STL-CQA: Structure-based Transformers with Localization and Encoding for Chart Question Answering

Hrituraj Singh
Adobe Research
hrisingh@adobe.com

Sumit Shekhar
Adobe Research
sushekha@adobe.com

Abstract

Chart Question Answering (CQA) is the task of answering natural language questions about visualisations in the chart image. Recent solutions, inspired by VQA approaches, rely on image-based attention for question/answering while ignoring the inherent chart structure. We propose STL-CQA which improves the question/answering through sequential elements localization, question encoding and then, a structural transformer-based learning approach. We conduct extensive experiments while proposing pre-training tasks, methodology and also an improved dataset with more complex and balanced questions of different types. The proposed methodology shows a significant accuracy improvement compared to the state-of-the-art approaches on various chart Q/A datasets, while outperforming even human baseline on the DVQA Dataset. We also demonstrate interpretability while examining different components in the inference pipeline.

1 Introduction

Charts Question Answering (CQA) (Kafle et al., 2018; Kahou et al., 2017; Chaudhry et al., 2020; Methani et al., 2020a) is the task designed on the lines of Visual Question Answering (VQA) (Antol et al., 2015; Malinowski and Fritz, 2014) which requires answering natural language questions about the data visualisations such as bar charts, pie charts, etc. The problem provides us with ability to understand charts using natural language queries, as well as grounding to the natural language statements for the reasoning operations being carried out to retrieve the final answer to the query.

CQA is a challenging task because of the following reasons - (a) large question/answer vocabulary due to chart-specific words, (b) Requirements of multi-modal *fine-grained* reasoning through understanding of natural language question as well as the visualizations. This is different from VQA, where the answer dictionary is typically limited, and the reasoning is coarse-grained as compared to that required for data visualisations, where finer details like bar length and color can heavily influence both the reasoning and the answer.

Despite data visualisations being ubiquitous in documents, the problem has received sparse attention in the literature. The earlier datasets like DVQA (Kafle et al., 2018) and FigureQA (Kahou et al., 2017) consist of charts generated from synthetic data, though there has been a push for data charts generated from real sources (Chaudhry et al., 2020; Methani et al., 2020a) as well. Due to the problems discussed above, the prior work noted that VQA algorithms cannot be applied directly to CQA. Hence, different CQA methods introduce modifications for the problem, while building on the backbone of VQA approaches. While FigureQA (Kahou et al., 2017) uses relational networks for question/answering, DVQA (Kafle et al., 2018) combines text detection and VQA-based attention modules to answer chart questions. LEAF-QA (Chaudhry et al., 2020) encodes question/answers in terms of chart elements, to handle infinite vocabulary problem, while resorting to a VQA-based model as the backbone.

Though the approaches improve performance for various chart datasets, the challenges of robust reasoning over varied chart varieties, are far from being solved. We posit that this is mainly due to the non-exploitation of the significant characteristics of charts that distinguish them from plain natural images - the structure and set of chart elements. The structure of the charts along with the position of different chart elements must be exploited by the learning models to enable reasoning over them from natural language questions. In this paper, we propose a transformer-based model to exploit such structural properties of data visualisations, while

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 3275–3284,
November 16–20, 2020. ©2020 Association for Computational Linguistics

also showing that our model can provide a much deeper and better interpretations to the generated answers. Our key contributions can be summarized as follows:

- We propose a transformers-based framework to *fully* utilize the structural properties of charts and achieves state-of-the-art performance on the task of charts question answering.
- We define a set of pre-training tasks for inducing structural knowledge of charts or data visualisations into the proposed model and demonstrate its effectiveness.
- We conduct a range of interpretability experiments to dissect the reasoning process of our model.
- We extend the recently proposed LEAF-QA dataset (Chaudhry et al., 2020) to generate a *harder* and *more* balanced dataset.

2 Related Works

Visual Question Answering: The problem of Visual Question/Answering (VQA) has been explored extensively with a variety of datasets (Malinowski and Fritz, 2014; Antol et al., 2015; Ren et al., 2015; Krishna et al., 2017; Kafle and Kanan, 2017) with various approaches for joint understanding of images and text. A more closely related work to our problem is, however, TextVQA (Singh et al., 2019) which focuses on the problem of question/answering with scene texts, having *infinite* vocabulary. Correspondingly, a variety of solutions have also been proposed - the most successful have been based on attention (Xu et al., 2015; Yang et al., 2016; Anderson et al., 2018) and joint multimodal learning (Tan and Bansal, 2019; Lu et al., 2019; Chen et al., 2019; Li et al., 2020).

Pre-training: The success of pre-training with ELMo (Peters et al., 2018), GPT (Radford et al., 2018), GPT-2 (Radford et al., 2019), BERT (Devlin et al., 2019) has led to significant advancements in natural language understanding. These pre-training frameworks also motivated some of the recent works on multi-modal understanding (Tan and Bansal, 2019; Lu et al., 2019; Chen et al., 2019; Sun et al., 2019). Our pre-training framework borrows ideas from these works with additional tasks designed specifically for understanding chart structure. To the best of our knowledge, ours is one of the first work to demonstrate the effectiveness of pre-training in inducing structural knowledge of charts.

Charts Questions Answering: There has been several works lately addressing the problem of CQA. One line of work relies on using the chart figures and questions directly (Kahou et al., 2017; Kafle et al., 2018, 2020; Chaudhry et al., 2020) while others (Methani et al., 2020a; Qian et al., 2020) are focused on parsing out the chart data first to perform the task. Our approach falls in the first category. Apart from chart question answering, there have been works focused on chart data parsing (Cliche et al., 2017; Kallimani et al., 2013; Savva et al., 2011) or visual structure extraction (Tsutsui and Crandall, 2017; Poco and Heer, 2017). While they do not focus on natural language based understanding of charts, their components form the basis for our structural understanding of charts.

3 STL-CQA

In this section, we describe our overall framework which is the first method to *fully* utilize the structural knowledge of charts for both question encoding and reasoning to perform the task of Charts Question Answering (CQA). We refer to our framework as STL-CQA - **S**tructure-based **T**ransformers with **L**ocalization and encoding for CQA. Even though prior works have attempted to utilize the chart structure for encoding questions, their reasoning frameworks still do not exploit this knowledge resulting in sub-optimal performances and offering much less insight into the reasoning process of these models. We divide our overall framework into three stages - Localization, Encoding, and Transformers-based structural attention. While the first two stages have been adopted from the existing state of the art frameworks, the novel reasoning stage makes our algorithm much more powerful and interpretable as discussed in the later sections.

3.1 Localization

The first step in our multi-stage framework is the detection or localization of the chart elements used in different types of data visualisations. For this purpose, we leverage the advances in the object detection frameworks and use the Mask-R CNN (He et al., 2017) with a Resnet-101 backbone. We enlist the different categories of our elements and train the network from scratch on the training sub-

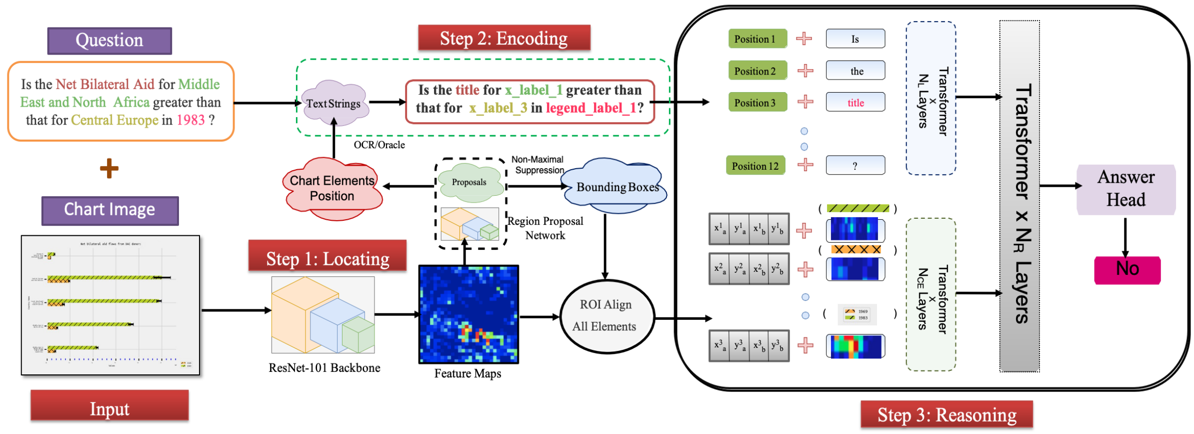


Figure 1: Overview of our pipeline showing the three different stages of our overall pipeline. We first encode both the question string and chart image by locating the different chart elements. The reasoning module, then, processes both the encoded question as well as the encoded chart elements structurally in order to provide the final answer.

set (described in Section 4.1) of around 198K images. Since the metadata provided in the public datasets such as DVQA (Kafle et al., 2018)/LEAF-QA (Chaudhry et al., 2020)[1] consist of only bounding boxes, we convert them into masks using several approximations specially for pie/donut charts where we utilize the geometry of different figures to prepare masks (refer supplementary for details). The implementation is carried out through Detectron2 (Wu et al., 2019) framework with a learning rate initialization of 0.00025 for 150, 000 iterations.

3.2 Encoding

Unlike VQA, the text vocabulary in the case of CQA is much larger if not *infinite*. For each chart, a question about it consist of words whioch are very specific to that chart. For example - A chart showing GDP of different countries can have words like 'USA' or 'Canada' which might not be present in other charts at all. We, therefore, follow dynamic encoding scheme (Chaudhry et al., 2020; Kafle et al., 2020) to encode the questions. In this paper, we only report performance with a text oracle, which is same as the previous work (Kafle et al., 2020; Chaudhry et al., 2020). The oracle is a perfect OCR which provides access to the bounding boxes and content of different text areas on charts, while the role of the text area (`x-title, y-title, etc.`) is taken from our localization system. We use the bounding box information to assign the relative *position* to each of the text string. The positioning scheme (shown in Fig 2) is based on (Chaudhry et al., 2020) where `x-axis` labels are assigned positions in increasing order from left-right, `y-axis` labels and `legend` labels are assigned *positions* bottom-top and (left-right, top-bottom) respectively. For pie charts and donut charts, the *positions* are assigned in an anti-clockwise manner.

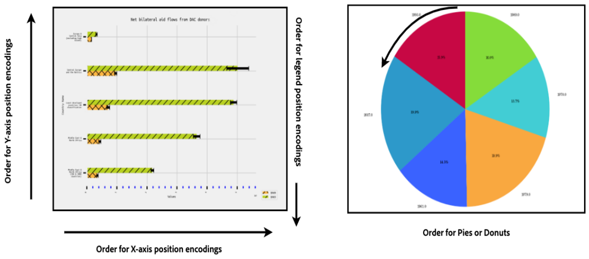

Order for Y-axis position encodings
Order for legend position encodings
Order for X-axis position encodings
Order for Pies or Donuts

Figure 2: Position Encoding Scheme Used for encoding the chart strings as well as different chart elements.

The extracted strings and their *positions* are then used to replace the string of the question with standardized tokens. For example - if the token in the question string is 'USA' which is present as an `x-axis` label as its third element from the origin, we replace the token 'USA' with `xlabel_3`. The vocabulary of questions, thus, consists of both standardized tokens as well as natural language tokens such as *greater*, *which* etc. which are common to all questions. The vocabulary (or classes) of answers is determined in the exact similar manner.

3.3 Structure-based Transformers

This is the novel and most important module of our framework which performs **(a)** chart structure

[1] Another popular dataset FigureQA does not provide the bounding box

understanding, **(b)** question understanding, and **(c)** reasoning over the chart to find the answer. We adapt the transformer-based frameworks from (Tan and Bansal, 2019; Lu et al., 2019; Chen et al., 2019; Li et al., 2020) to perform reasoning over charts. We demonstrate, empirically, that the architecture is strongly suited for the task of CQA through extensive experiments. The architecture can be broken down into 4 stages:

Input: The inputs to the model are two sequences of features. The question Q is broken down into a sequence of words $\{w_0, w_1, ..., w_n\}$ and encoded as a sequence of word embeddings $\{e_0, e_1, ..., e_n\}$ of dimension d_e also taking the position into account:

$$e_i = \text{word-emb}(w_i) + \text{pos-emb}(i) \qquad (1)$$

A normalisation layer is applied before providing the word embedding sequence as input to our model.

For the chart image C, the model input is prepared by utilizing the output of Mask-RCNN (Anderson et al., 2018). We extract the features using the Resnet-101 backbone of our detection network and use the bounding boxes of different m chart elements $\{c_0, c_1, ..., c_m\}$ as well to encode the chart. Although natural images have larger number of possible class elements, in the case of data visualizations, more class elements are present simultaneously in an image. Further, reasoning in charts depends heavily on the correct detection of the geometry and type of each box. Hence, unlike (Tan and Bansal, 2019), where a fixed number of objects are extracted for every image even if there are several overlaps, we apply non-maximal suppression (Neubeck and Van Gool, 2006) to choose the most confident and distinct bounding boxes. Finally, the Resnet-101 network is used to extract the features of the final bounding boxes. A plot class which provides a bounding box of the plot region to provide a global picture, is also taken. This is necessary for answering the global information questions about the images (such as *Is there a grid in the chart?*). We found the performance to improve significantly after adding the global plot representation. Since, different images can have different number of chart elements, we pad the sequences to have a fixed length M for all charts. The chart input sequence is computed as below:

$$f_i = \textbf{LayerNorm}(W_F r_i + b_f) \qquad (2)$$

$$p_i = \textbf{LayerNorm}(W_P x_i + b_p) \qquad (3)$$

$$c_i = \frac{f_i + p_i}{2} \qquad (4)$$

where, r_i corresponds to the Resnet-101 features of i^{th} chart element, x_i refers to corresponding bounding box coordinates, (W_F, b_f) and (W_P, b_p) are learnable parameters.

Chart Relation Transformer: The chart features computed in Eq. 4 are fed to a transformer with N_{CE} layers each having a self-attention block and a feed-forward block both with residual connections as proposed originally by (Vaswani et al., 2017) . The chart-only transformer learns relationships between chart elements, agnostic of the question. We discuss more details about these relationships and their interpretation in Section 4.4.

Question Transformer: This is again a transformer with N_L layers each having a self-attention block and a feed-forward block with residual connections to encode the meaning of the question. The input to the encoder is as computed in Eq. 1. We also tried using token IDs along with positional embedding to distinguish between words from common vocabulary (eg. how, many) and standardized words for chart vocabulary (e.g. `xtitle`, `legend_title`). but it did not yield any further improvement.

Reasoning Module: The reasoning module is the cross-attention transformer block with N_R layers which takes as input the contextual features generated by chart transformer and question transformer. Each layer consists of three blocks - cross-attention, self-attention, and feed-forward. In the cross-attention block of chart stream, chart features act as *query* in the attention formulation (Bahdanau et al., 2014) and the features from question *stream* act as *keys* as well as *values* while the vice-versa happens in the cross-attention block of question stream. This block is followed by a self-attention block and a feed-forward block acting independently in their own streams. All of the three blocks have residual connections. If the i^{th} question token's features and j^{th} chart element's features being used as input for k^{th} layer are represented by

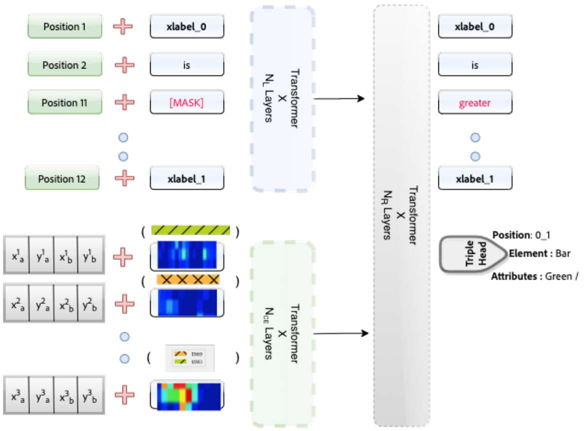

Position 1
xlabel_0
Position 2
is
Position 11
[MASK]
Position 12
xlabel_1
Transformer X N_L Layers
x^1_a y^1_a x^1_b y^1_b
x^2_a y^2_a x^2_b y^2_b
x^3_a y^3_a x^3_b y^3_b
Transformer X N_{CE} Layers
Transformer X N_R Layers
xlabel_0
is
greater
xlabel_1
Triple Head
Position: 0_1
Element : Bar
Attributes : Green /

Figure 3: Overview of our pre-training task. MLM is used to recover the language tokens thus inducing both language structure and cross-modality understanding. NSP-like task is also used for the same properties. The triple head on each vision element predicts the position, the chart element class or category, and its attributes.

Q^i_{k-1} and C^j_{k-1} and attention with q query, k keys, and v values is represented by $attn(q, k, v)$ then cross attention block for question stream can be represented as in Eq. 5 and self-attention block as in Eq. 6

$$Q^i_{k_{cross}} = attn(Q^i_{k-1}, C_{k-1}, C_{k-1}) \qquad (5)$$

$$Q^i_{k_{self}} = attn(Q^i_{k_{cross}}, Q_{k_{cross}}, Q_{k_{cross}}) \qquad (6)$$

where $Q_k : \{Q^0_k, ..., Q^n_k\}$ and $C_k : \{C^0_k, ..., C^m_k\}$.

We show the reasoning module as a single large block in Fig. 1. Since the cross attention module is followed by self-attention module for each layer, the information (what is asked) from the cross-attention is used by the self-attention layers to perform various operations with each other. We discuss more about the reasoning operations carried out by the model in Section 4.4. The **[CLS]** token prepended to the question tokens captures the entire cross-modal information and is used to retrieve the final answer by applying a two layer perceptron over the contextual embedding for this token. The whole system is trained using a cross-entropy loss.

3.4 Pre-training

In this section, we propose a set of pre-training tasks for our STL-CQA model. To the best of our knowledge, this is the first use of pre-training for charts question answering. Our proposed tasks are on the lines of pre-training literature in language modelling (Devlin et al., 2019) and VQA (Tan and Bansal, 2019). Our tasks can be primarily grouped into three categories:

Chart Structure tasks consists of the tasks designed to induce the sense of different parameters which make up the properly defined structure of the chart. We focus on three major things - (a) Types of chart elements (b) Position of chart elements (c) Color and pattern of non-textual elements in charts. Unlike (Tan and Bansal, 2019; Lu et al., 2019), we do not pre-train our model on the features regression task. For the type of chart elements, we consider 23 chart categories and use a cross-entropy classification loss for each element over them. For the position of chart elements, we use positioning scheme similar to the one discussed in Section 3.2. Since, even along x-axis (or y-axis in case of horizontal graphs), we can have multiple groups, we use a positioning scheme for chart elements as well. For example, a stacked bar chart having a bar at third position on x-axis (left to right) and second position in legend box (top to bottom) is assigned a position 2_1 (zero-indexing). These positions are then treated as targets for a classification task using a linear position head like that for types of charts elements. For colors and patterns, we use the chart metadata. We treat a particular color and pattern combination as a category and train the model on identifying the color and patterns as a classification problem.

Language/Question: For language tasks, as is prevalent in recent works, we train the model on standard MLM i.e. Masked Language Modelling task (Devlin et al., 2019) task. However, in our case, we do not just randomly mask any word. We specifically focus on chart vocabulary words or words which modify the meaning of the sentence such as *higher*, *lower* etc. During caption generation, we keep track of such words and pass their indices to random masking function so that only those indices are masked during the training stage.

Cross Modal: For the reasoning module, we use only one pre-training task which is similar to the next sentence prediction task of BERT. We replace the original sentence with a mismatched sentence with a probability of 0.5 and then train a classifier to identify the mismatched sentence.

4 Experiments

We conduct a range of experiments to demonstrate the efficacy of our proposed STL-CQA network. In this section, we describe the different datasets which were used along with the models and the obtained results.

4.1 Dataset

Split	Structure	Data	Reasoning	Overall
Train	605,176	715,697	742,588	2,063,461
Test-Familiar	120,884	143,029	148,589	412,502
Test-Novel	35,094	40,611	37,687	113,392

Table 1: Numbers of questions by type for the LEAF-QA++ Corpus.

We evaluate the proposed STL-CQA method on recent chart question/answering datasets. DVQA (Kafle et al., 2018) has a large corpus of bar charts and associated question/answers. We use the splits as provided in (Kafle et al., 2018) for our experiments. We demonstrate that the proposed STL-CQA method outperforms the prior baselines.

LEAF-QA (Chaudhry et al., 2020), is a comprehensive chart question/answering dataset, covering 10 different types of charts and over 35 question templates. Using the publicly available chart annotations[2], we further develop a more comprehensive question/answering corpus, LEAF-QA++. The original LEAF-QA utilizes automatic paraphrasing of questions to generate variations. We manually curate 3-8 paraphrase variations of question templates to greatly increase the diversity and naturalness of the questions. We further, add new data question types, increasing the number of template questions from 35 to 75. We add data questions, not present in the original corpus, which ask about chart component positional or values. The question set is balanced to avoid pre-dominant values in answers, especially for questions with common chart answers (like yes/no). We refer the reader to the supplementary for further details on the proposed LEAF-QA++ corpus.

To prepare the data for pre-training, we generate 35 sentence templates for LEAF-QA++ using the metadata. Such templates are also augmented with a small list for each sentence which provides information about which are the relevant tokens for MLM masking. For each template, we use paraphrases which are written manually and also combine it with the templates of one-another with a probability of 0.5 thus producing a very high number of combinations.

4.2 Model Settings

For all the experiments, we use $N_{CE} = 5$, $N_L = 4$, and $N_R = 5$. We use 4 layers in language model, as the template-based questions even with paraphrasing, are less complex than the natural language. In fact, increasing the number of layers resulted in a deteriorated performance as the model overfitted to the vocabulary. For element relationship and reasoning blocks, we set N_{CE} and N_R to be 5 layers each. We use $d_e = 2048$ for consistency, the maximum length for questions is 30 and the maximum number of chart elements is taken to be 45.

Pre-training Details: We use 23 object categories, 5301 color and patterns combinations for attributes, and 63 different position combinations. We pre-train the model for 4 epochs on 4 V100 GPUs using an Adam Optimizer (Kingma and Ba, 2014) with an initial learning rate of $5 * 10^{-5}$ and batch size of 512.

Fine Tuning Details: We fine-tune the model for 6 Epochs if it has been pre-trained or for 10 epochs if the model is being trained from the scratch. The batch size used is 512 and an Adam optimizer is used with an initial learning rate of 10^{-4}.

4.3 Results

We show results on two datasets - DVQA and LEAF-QA++. We do not show our results on the LEAF-QA corpus as LEAF-QA++ is a superset of it. As discussed in Section 3.2, we assume access to an oracle in our experiments. We show comparisons with the current state-of-the-art models on these datasets. For DVQA comparison, we enlist the results from prior models, viz. QUES, IMG+QUES and SANDY (Kafle et al., 2018), PReFIL (Kafle et al., 2020), Plot-QA (Methani et al., 2020b). As shown in Table 3, both STL-CQA and PreFIL outperform human baselines. STL-CQA further improves over PReFIL, specially in the complex reasoning questions. For LEAF-QA++, we use the LEAF-Net model, the state-of-the-art on LEAF-QA and train it with the hyper-parameters mentioned in (Chaudhry et al., 2020). As discussed in (Chaudhry et al., 2020), previous models trained on DVQA are not directly applicable to LEAF-QA, due to the higher complexity of charts in the latter. Our model shows a significant improvement in ac-

[2] https://chartinfo.github.io/

Structure	**Data**	**Reasoning**
What type of graph is this ?	*What does the i bar from left in each group represent ?*	*Between* `legend_label_i` *and* `legend_label_i` *, which has higher* `ytitle` *for* `xlabel_i` *?*
Is there a grid in this graph ?	*Does the value of* `legend_label_i` *monotonically increase over* `xtitle` *?*	*Does there exist any* `xtitle` *where* `legend_label_i` *has higher* `ytitle` *than* `legend_label_i`*?*
Is there a legend in this graph ?	*How many groups or stacks of bars have ratio less than 2 between highest and lowest value bars ?*	*In what* `xtitle` *is the sum of* `legend_label_i` *and* `legend_label_i` *lower than* `legend_label_i` *?*
How many labels are there in the legend ?	*In or at which* `xtitle` *does* `legend_label_i` *have the highest* `ytitle` *?*	*In or at which* `xtitle` *does* `legend_label_i` *and* `legend_label_i` *have the highest difference?*

Table 2: Question samples of different types in LEAF-QA++ corpus.

Baselines	**Test-Familiar**				**Test-Novel**			
	Structure	**Data**	**Reasoning**	**Overall**	**Structure**	**Data**	**Reasoning**	**Overall**
QUES	44.03	9.82	25.87	21.06	43.90	9.80	25.76	21.00
IMG+QUES	90.38	15.74	31.95	32.01	90.06	15.85	31.84	32.01
SANDY	96.47	65.40	44.03	56.48	96.42	65.55	44.09	56.62
Plot-QA	-	-	-	57.99	-	-	-	59.54
LEAF-Net	98.42	81.25	61.38	72.72	98.47	81.32	61.59	72.89
Human	-	-	-	-	96.19	88.70	85.83	88.18
PReFIL	99.77	95.80	95.86	96.37	99.78	96.07	95.99	96.53
STL-CQA	**99.79**	**95.92**	**97.60**	**97.35**	**99.78**	**96.10**	**97.77**	**97.51**

Table 3: Results of comparison for different methods on familiar test and novel test subsets of DVQA.

Baselines	**Structure**	**Data**	**Reasoning**	**Overall**
QUES (ENC)	35.58	33.12	43.56	37.60
IMG	11.44	6.67	1.4	6.19
LEAF-Net	80.57	49.75	51.16	58.34
STL-CQA (w/o pre-train)	**93.12**	**89.12**	**88.97**	**90.24**
STL-CQA (Pre-trained)	**94.28**	**91.38**	**91.32**	**92.22**

Table 4: Result over Test-Familiar Split For LEAFQA++ Dataset.

Baselines	**Structure**	**Data**	**Reasoning**	**Overall**
QUES (ENC)	36.42	31.97	42.93	36.99
IMG	8.64	7.51	1.8	5.96
LEAF-Net	74.24	47.26	50.96	56.84
STL-CQA (w/o pre-train)	**88.34**	**76.92**	**82.95**	**82.46**
STL-CQA (Pre-trained)	**89.96**	**78.67**	**85.82**	**84.54**

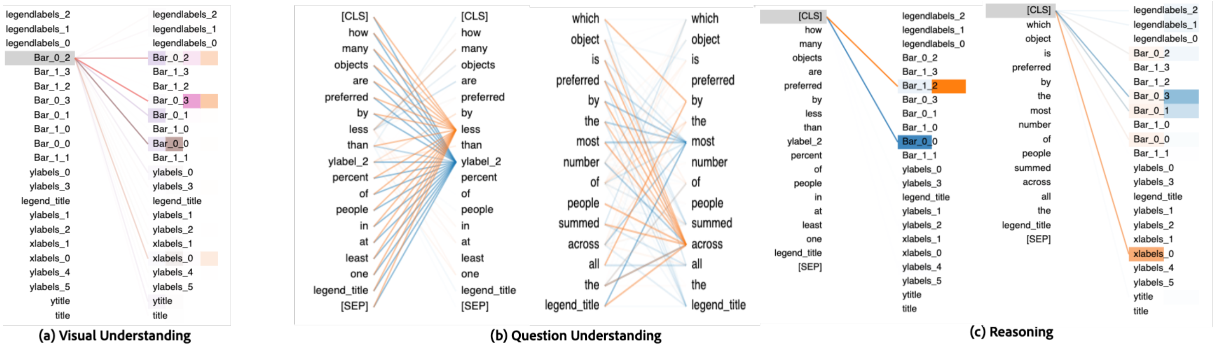
Table 5: Result over Test-Novel Split For LEAF-QA++ Dataset.

curacy over LEAF-Net with an overall increase of over 28%. The improvement is particularly remarkable for data and reasoning questions, showing that the VQA-based image attention network used in LEAF-Net do not generalize well for complex questions. The models on DVQA datasets have been able to outperform human baseline with significant margins which points towards the capability of the algorithms in performing crisp and consistent reasoning as compared to humans who are prone to data interpretation errors resulting in inconsistency, despite having better cognitive capabilities than the algorithms.

Pretraining: After pre-training , we fine-tune the model on our training subset for 6 epochs. While pre-training does help in score improvement across all three categories (Table 4 and 5), there is still a part of structural knowledge that the model fails to capture. Better and more focused pre-training tasks coupled with improved detection systems and larger datasets could help solve these problems in future.

4.4 Interpretability

One of the advantages of using structure-based interpretable elements in key, query and value of attention is the ease in grounding the attention weights to chart structure. In our case, each element in the language stream is a discrete token and elements in the visual stream correspond to chart elements. Thus, we are able to dissect the attention heads and interpret the semantic grounding of the attention weights. We isolate a single chart image with two questions in Fig. 4 to demonstrate the functions of three separate blocks as discussed in Section 3.3. Chart structure understanding is carried out with the visual understanding block. In this case, attention visualisa-


legendlabels_2
legendlabels_1
legendlabels_0
Bar_0_2
Bar_1_3
Bar_1_2
Bar_0_3
Bar_0_1
Bar_1_0
Bar_0_0
Bar_1_1
ylabels_0
ylabels_3
legend_title
ylabels_1
ylabels_2
xlabels_1
xlabels_0
ylabels_4
ylabels_5
ytitle
title
[CLS] how many objects are preferred by less than ylabel_2 percent of people in at least one legend_title [SEP]
which object is preferred by the most number of people summed across all the legend_title
[CLS] which object is preferred by the most number of people summed across all the legend_title [SEP]

(a) Visual Understanding

(b) Question Understanding

(c) Reasoning

Figure 4: Interpretability over the steps performed to recover the answer. Color Coding denotes heads. For the same chart, visual understanding remains same. We show visualisation using certain selected heads depending upon the function for visual understanding while limiting to two heads only (max values) for Question and Reasoning

tion shows that it is organising the grouped bar chart into families on the basis of their group. `Bar_0_2` means a bar from group at `xlabels_0` present at 2^{nd} position from left (for a vertical chart) and its attention is linked to the other bars in this same group, (`Bar_0_0`, `Bar_0_1`, `Bar_0_2`, `Bar_0_3` and the class of that group `xlabels_0`). We find these heads to be consistent even for other bars. We also find some attention heads establishing relationship between those bars which are from the same legend group. The question understanding visualisations (for two specific heads of last layer) for first question show a heavy focus on the two important parts of the question, contributing to determination of the answer i.e. less and `ylabel_2` with some focus on 'how many' which determines that this is a *counting* question. The language understanding visualisation for these two heads for the second question also shows similar functions for them.

The last layers of the reasoning block for first question shows **[CLS]** token (which is used in the answer head) putting almost all its attention in two bars. We find these two bars to be the one satisfying the criteria of being 'less than `ylabel_2`'. The answer of this question ('two') is predicted correctly by the model. For the second question **[CLS]** token puts all almost all its attention on `xlabel_0` which is the correct answer while putting some attention on the bars which are contributing to the sum. Infact, the second highest attention is on the bar having highest value.

5 Discussion and Limitations

While our proposed model is able to reason very effectively achieving state-of-the-art on the recent datasets, it is able to do so with an assumption of perfect OCR. Thus, it will be pertinent to have better OCR models for chart images. While reasoning in a fine grained manner has been an important part of CQA, the proposed STL-CQA method shows that reasoning could be performed with a high accuracy, given the elements of the charts have been detected accurately.

Even though our model achieves near perfect accuracy on the public datasets, the current datasets are synthetic and may not represent the plethora of chart visualisation styles used in real life. Despite the significant progress in simulating real world chart understanding scenarios, especially in LEAF-QA++, there are underlying biases in the generation process (for e.g. due to using a single software like `Matplotlib` for generation). However, we believe that these are important steps towards the eventual goal of understanding charts in the wild.

A further limitation is that the questions used in the existing datasets are template-based. Even though we make an attempt in LEAF-QA++ to increase the number of templates as well as manual generation of paraphrases to bring more diversity, the current templates do not capture the full range of variations in the questions which can be asked from the visualizations. The manually generated questions will also bring different ways to address the same text on chart images. For example - Gross Domestic Product could be addressed with its more common short form, 'GDP'. The current approach relies on text-string matches to encode questions, and works because the questions have been generated using the original chart text strings. This approach, will however fail in scenarios where the entity could be addressed through its variations. However, bringing human-generated questions into the proposed datasets is a challenge since human subjects would be required to possess a deep understanding of the different charts, before being able to ask reasonable and difficult questions.

6 Conclusion and Future Works

In this work, we proposed an extension to the LEAF-QA data using the public metadata provided for the charts. We also proposed a transformers-based framework while emphasizing on the need to exploit the structural properties of chart, and showed its strong effectiveness by achieving state-of-the-art with a significant margin on the recent Chart Q/A datasets. We also defined and experimented with a set of pre-training tasks and showed the improvement due to pre-training on the problem of CQA. We used attention to dissect our model to show how each of its module functions to retrieve the final answer. We discussed the current line of CQA work and proposed future directions by outlining the limitations of the current datasets and models.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

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

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations. *arXiv preprint arXiv:1909.11740*.

Mathieu Cliche, David Rosenberg, Dhruv Madeka, and Connie Yee. 2017. Scatteract: Automated extraction of data from scatter plots. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 135–150. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969.

Kushal Kafle and Christopher Kanan. 2017. An analysis of visual question answering algorithms. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1965–1973.

Kushal Kafle, Brian Price, Scott Cohen, and Christopher Kanan. 2018. Dvqa: Understanding data visualizations via question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5648–5656.

Kushal Kafle, Robik Shrestha, Scott Cohen, Brian Price, and Christopher Kanan. 2020. Answering questions about data visualizations using efficient bimodal fusion. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1498–1507.

Samira Ebrahimi Kahou, Vincent Michalski, Adam Atkinson, Ákos Kádár, Adam Trischler, and Yoshua Bengio. 2017. Figureqa: An annotated figure dataset for visual reasoning. *arXiv preprint arXiv:1710.07300*.

Jagadish S Kallimani, KG Srinivasa, and Reddy B Eswara. 2013. Extraction and interpretation of charts in technical documents. In *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 382–387. IEEE.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73.

Xiujun Li, Xi Yin, Chunyuan Li, Xiaowei Hu, Pengchuan Zhang, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. *arXiv preprint arXiv:2004.06165*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23.

Mateusz Malinowski and Mario Fritz. 2014. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in neural information processing systems*, pages 1682–1690.

Nitesh Methani, Pritha Ganguly, Mitesh M Khapra, and Pratyush Kumar. 2020a. Plotqa: Reasoning over scientific plots. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1527–1536.

Nitesh Methani, Pritha Ganguly, Mitesh M Khapra, and Pratyush Kumar. 2020b. Plotqa: Reasoning over scientific plots. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1527–1536.

Alexander Neubeck and Luc Van Gool. 2006. Efficient non-maximum suppression. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, pages 850–855. IEEE.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Jorge Poco and Jeffrey Heer. 2017. Reverse-engineering visualizations: Recovering visual encodings from chart images. In *Computer Graphics Forum*, volume 36, pages 353–363. Wiley Online Library.

Xin Qian, Eunyee Koh, Fan Du, Sungchul Kim, and Joel Chan. 2020. A formative study on designing accurate and natural figure captioning systems. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems Extended Abstracts*, pages 1–8.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Mengye Ren, Ryan Kiros, and Richard Zemel. 2015. Exploring models and data for image question answering. In *Advances in neural information processing systems*, pages 2953–2961.

Manolis Savva, Nicholas Kong, Arti Chhajta, Li Fei-Fei, Maneesh Agrawala, and Jeffrey Heer. 2011. Revision: Automated classification, analysis and redesign of chart images. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 393–402.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards vqa models that can read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8317–8326.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7464–7473.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5103–5114.

Satoshi Tsutsui and David J Crandall. 2017. A data driven approach for compound figure separation using convolutional neural networks. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 533–540. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. 2019. Detectron2. `https://github.com/facebookresearch/detectron2`.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. 2016. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 21–29.