Answering Questions about Charts and Generating Visual Explanations

Dae Hyun Kim
Stanford University
Stanford, CA, USA
dhkim16@cs.stanford.edu

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

Maneesh Agrawala
Stanford University
Stanford, CA, USA
maneesh@cs.stanford.edu

ABSTRACT

People often use charts to analyze data, answer questions and explain their answers to others. In a formative study, we find that such human-generated questions and explanations commonly refer to visual features of charts. Based on this study, we developed an automatic chart question answering pipeline that generates visual explanations describing how the answer was obtained. Our pipeline first extracts the data and visual encodings from an input Vega-Lite chart. Then, given a natural language question about the chart, it transforms references to visual attributes into references to the data. It next applies a state-of-the-art machine learning algorithm to answer the transformed question. Finally, it uses a template-based approach to explain in natural language how the answer is determined from the chart's visual features. A user study finds that our pipeline-generated visual explanations significantly outperform in transparency and are comparable in usefulness and trust to human-generated explanations.

Author Keywords

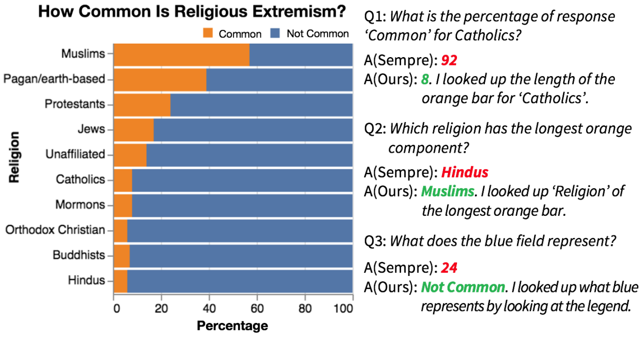
Question answering; Visualization; Explainable AI;

CCS Concepts

•**Human-centered computing** → **Natural language interfaces;**

INTRODUCTION

Using visualizations to analyze data, answer questions, and explain how the answer was obtained, is at the heart of many decision-making tasks. However, performing such complex analytical tasks with visualizations is not always easy. Users often need to answer compositional questions that require combining multiple complex operations such as retrieving a value from the chart, finding extreme values, comparing and aggregating values, or calculating sums and differences of values. Consider the bar chart in Figure 1 and the question *"For which religion did the most chaplains think that religious extremism is common?"* To answer this question, users need to visually compare the values represented by orange bars, find the longest one and then lookup the corresponding religion; in this case it is *'Muslims'*.



Q1: *What is the percentage of response 'Common' for Catholics?*

A(Sempre): ***92***
A(Ours): ***8****. I looked up the length of the orange bar for 'Catholics'.*

Q2: *Which religion has the longest orange component?*

A(Sempre): ***Hindus***
A(Ours): ***Muslims****. I looked up 'Religion' of the longest orange bar.*

Q3: *What does the blue field represent?*

A(Sempre): ***24***
A(Ours): ***Not Common****. I looked up what blue represents by looking at the legend.*

Figure 1. Questions about a chart from a Pew research report [3]. Q1 requires a value lookup on the data in the chart and Q3 requires a lookup on the legend. Q2 is compositional as it requires multiple operations including value lookup and comparisons. Our automatic chart question answering pipeline answers all three questions correctly (marked in green) and gives correct explanations of how it obtained the answer, whereas Sempre [43, 59], a state-of-the-art table question answering system, gets all three wrong (marked in red).

As users are analyzing a chart, they regularly pose questions by referring to visual features of the chart including the graphical marks (e.g. bars) and their data encoding visual attributes (e.g. width) [52, 53, 29]. For example, in the course of analyzing the bar chart in Figure 1, a user might ask *"Which religion has the longest orange component?"* This is a visual version of our earlier question, and while it remains compositional, because it references visual features of the chart, it is shorter and more directly suggestive of the operations users must perform to answer it. Nevertheless, answering such compositional questions, whether they are visual or non-visual, can be time-consuming and mentally taxing as users must perform multiple complex operations.

To obtain a better insight into how people naturally ask questions about charts, we conducted a formative study in which we collected 629 human-generated questions for 52 real world charts along with 748 human-generated explanations. We then categorized the questions along two orthogonal dimensions; (1) *lookup* (i.e. requiring a single value retrieval), or *compositional* (i.e. requiring multiple operations) and (2) *visual* (i.e. referencing visual chart features) or *non-visual*. We find that people frequently ask compositional questions (70%), regularly ask visual questions (12%), and that visual explanations are especially common (51%).

Can we design a tool to automatically answer such natural language questions about charts? Automatic question answer-

CHI '20, April 25–30, 2020, Honolulu, HI, USA.

ACM ISBN 978-1-4503-6708-0/20/04.
http://dx.doi.org/10.1145/3313831.3376467

ing would benefit users in several ways. It would significantly reduce the time and mental effort by performing complex operations such as retrieval, comparison and aggregation (sum, average) on behalf of users. Such a tool could quickly and accurately retrieve data values from visual attributes that are perceptually difficult to decode (e.g. size, brightness). More importantly, introducing a natural language interface into the data analysis workflow would lower the threshold of ability required to analyze data using charts and graphs. It could enable people who have not been formally trained in data analysis tools and visualization literacy to get answers to their questions. However, for users to rely on such an automated tool, it is critical that the tool be able to transparently explain how it obtains the answers [52]. Moreover, our formative study suggests that the most effective explanations are visual because they describe how the answer is extracted from the visual features of the chart. Yet, no previous work on automatic question answering for charts [29, 27, 14, 47, 28] has provided explanations for their answers.

In this paper, we present an automatic pipeline for answering natural language questions about charts and generating such visual explanations. Our approach builds on Sempre [43, 59], a question-answering system for relational data tables that focuses on answering compositional, non-visual questions. We significantly extend Sempre to answer questions about charts and also generate corresponding visual explanations. Our pipeline works with both lookup and compositional questions as well as visual and non-visual questions. The key idea of our approach is to take advantage of the visual data encoding structure of an input chart in Vega-Lite [50] format – a programmatic representation that explicitly describes the encodings that map data to mark attributes – to accurately answer visual questions and to generate the visual explanations.

We evaluate our question-answering pipeline on the corpus of 629 chart-question pairs we gathered in our formative study. We find that our pipeline correctly answers 51% of all the questions in our corpus, while Sempre alone can only answer 39% of the questions correctly, a difference of 12%. For visual questions, our improvement is even larger at 53% and even for non-visual questions, our pipeline outperforms Sempre by 6%. Overall, these results suggest that information about the visual encoding structure of a chart is very useful for automatic chart question answering. Finally, we conduct a user study which finds that our pipeline-generated visual explanations are significantly more transparent than human-generated explanations while remaining comparable in usefulness and trust.

RELATED WORK

Our work is grounded in three main areas of prior work; (1) Natural language interactions with visualization, (2) Automatic question answering and (3) Explainable AI.

Natural Language Interactions with Visualization

Natural language interfaces for visualizations have received considerable attention recently. Typically, these interfaces respond to natural language queries by either creating a new visualization (e.g. DataTone [22]) or by highlighting answers within an existing visualization (e.g. Eviza [52], Evizeon [25]). Some systems enable follow-up queries with support for pragmatics to consider context from past queries [17, 57, 18, 25]. Other systems give users feedback describing how the system interprets a query and allow users to correct misunderstandings [22, 52, 25, 54]. However, all of these systems focus on developing interfaces that work with simple natural language queries to generate visual output. In contrast, our work focuses on developing algorithms for answering visual and compositional natural language questions with text output.

A few researchers have focused on automatically connecting charts with text that refers to it in the surrounding document [31, 30, 11]. Kong et al. [31] present a crowdsourcing framework for extracting such references between surrounding text and chart elements as well as an interactive document browser that visualizes the references. Kim et al. [30] present a fully automated algorithm for finding references between surrounding text and a table as well as an interactive document viewer that similarly highlights the extracted references. Badam et al. [11] automatically parse documents to identify various text-based similarity relationships (e.g. stylistic similarity, semantic similarity) between tables and the surrounding text. In contrast to such natural language reference finding, our work focuses on answering natural language questions and explaining how the answers were determined.

Researchers have also attempted to automatically generate captions for charts [40, 15]. For example, Mittal et al. [40] apply a planning-based approach to generate captions. Chen et al. [15] propose an attention-based mechanism for generating natural language captions for charts. Instead of explaining the chart itself, our pipeline explains how it uses the chart to answer the question.

Automatic Question Answering

Automatic question answering using recent advances in machine learning [21] has been investigated in a variety of data domains, ranging from document collection [45, 58, 55] to data tables [43, 59, 34, 9, 60] to image collections [10, 36, 29, 27]. Question answering with data tables is the most relevant research to our work, since the underlying data of the vast majority of visualizations can be represented as tables. Sempre [43, 59] is a method for answering compositional questions with semi-structured tables. Their system translates the natural language question into a logical query and executes the query on the table to generate the answer. Researchers have also addressed the table question answering task by applying neural networks [32] and reinforcement learning techniques [34, 9, 60]. In our work, we adapt the semantic parsing model of Sempre [43, 59] to our problem in which questions regularly refer to visual features (e.g. marks and their visual attributes) of charts. We also extend this work to provide explanations for the answers it produces.

Question answering with images is also an active research topic [10]. While much of this work focuses on answering natural language questions about photographs, a few research groups have developed question answering techniques for visualizations and scientific figures [29, 27, 14, 47, 28]. All of these systems treat the input chart as an image and focus on applying computer vision techniques to obtain the answer.

In contrast, we consider the structure of the chart (e.g. how it encodes the data) to answer to the natural language question and to provide an explanation for it. Unlike existing systems, we also develop a corpus of charts from real-world sources and we gather crowdsourced natural language questions-answer pairs for each of them.

Explainable AI

While AI and machine learning have seen dramatic breakthroughs in the last few years, they usually produce black-box models; it is difficult to understand why or how a model makes a particular decision or produces an answer. Recent surveys document a variety of techniques designed to address the lack of interpretability of machine learning models [19, 7, 42]. They suggest that an explainable model describes the relationship between the system's input and its output; it can describe the mechanisms through which it makes decisions [7].

Researchers have focused on either providing instance-based explanation for a single prediction [56, 48, 49, 8] or explaining the overall behavior of the predictor [12, 20]. For example, saliency maps provide instance-based visual explanations of image classifiers by highlighting regions in an input image determining the output of a convolutional neural network classifier [56]. Lei et al. [33] present a method for sentiment prediction in review text, where it automatically extracts phrases from the review that leads to a particular prediction score. Our work also focuses on instance-based explanation as it describes how it produces an answer given an input question-chart pair. However, rather than extractive methods that highlight part of the input data, we develop a template-based natural language generation method that explains how the model reaches the answer by analyzing the chart. Our system is the first to generate explanation for chart question answering.

FORMATIVE STUDY

To learn how people naturally ask questions, extract answers and explain their answers when they encounter charts, we conducted a formative study. We gathered a corpus of charts from multiple real-world sources, and asked crowdworkers to write natural language questions, provide answers and explain their answers. We then manually analyzed the resulting data to understand (1) how often people ask lookup and compositional questions and (2) how often they refer to visual features for the charts in their questions and explanations.

Gathering Charts, Questions, Answers and Explanations

Our corpus includes 52 charts, gathered from four different sources; (1) the Vega-Lite Example Gallery [5], (2) charts in Pew Research Reports as collected by Kong et al. [31], (3) D3 charts we found across the Web, and (4) charts constructed from tables found in the WikiTableQuestions dataset [43]. In total, our corpus includes 47 bar charts (32 simple, 8 grouped, 7 stacked) and 5 line charts. We focus on these two chart types because, as Battle et al. [13] have shown, they are two of the most common types of charts available on the Web.

We asked crowdworkers from Amazon Mechanical Turk to consider a single chart, write 5 natural language questions about it, answer 10 questions about it including their own and provide explanations for their answers. We then manually reviewed the responses and removed questions that were not answerable from the chart, as well as explanations that carried no information about how the worker obtained the answer from the chart (e.g. *"I got it from the chart"*). This process generated a total of 629 questions, 866 answers and 748 explanations for the 52 charts.

	# Questions			# Explanations
	Lookup	Compositional	**Total**	
Visual	52 (8%)	24 (4%)	**76 (12%)**	380 (51%)
Non-Visual	138 (22%)	415 (66%)	**553 (88%)**	368 (49%)
Total	**190 (30%)**	**439 (70%)**	**629**	**748**

Table 1. Counts and percentages of the types (lookup/compositional, visual/non-visual) of natural language questions and explanations crowdworkers generated for our set of 52 charts.

Analysis

We analyzed the crowdworker responses to differentiate compositional questions from lookup questions as well as visual versus non-visual questions and explanations (Table 1).

We find that 70% of the questions are compositional, while the remaining 30% are lookups. Compositional questions often ask about extrema (38%), differences between two data values (22%), and the sum of multiple values (7%). An additional 12% of the compositional questions require performing multiple compositional operations to arrive at the answer (e.g. difference of the maximum and the minimum). People also regularly ask visual questions (12%) that refer to visual features of the chart. Visual questions tend to be lookups (68%) while non-visual questions tend to be compositional (75%).

Most importantly, we find that people frequently provide visual explanations (51%), which describe the process of extracting an answer from the visual features of a chart. Consider Q2 in Figure 1 where the correct answer is *’Muslims’*. A person might visually explain how they got the answer by reporting *"Muslims have the longest orange bar in the chart."* In contrast, a non-visual explanation such as *"Muslims are about 57% Common, more than any other Religion,"* only refers to the data and does not describe the process of extracting the data from the visual features of the chart. Thus, it is less thorough and this lack of completeness may explain why non-visual explanations are slightly less common than visual explanations.

Additional Collection of Visual Questions

To better understand how visual questions are posed we also collected a set of 277 visual questions about charts from our collegues. We analyzed these questions to to identify the lexical and the syntactic structures that people typically use to refer to marks and visual attributes in visual questions. We use the results of this analysis for converting visual questions to non-visual questions in Stage 2 of our pipeline.

METHOD

Our question answering system takes a chart and a natural language question as input and outputs the answer to the question along with an explanation (Figure 2). Our approach is to adapt Sempre [43, 59], a question answering algorithm that works with relational data tables instead of charts. In Stage 1 of our pipeline, we extract the visual encodings that map data

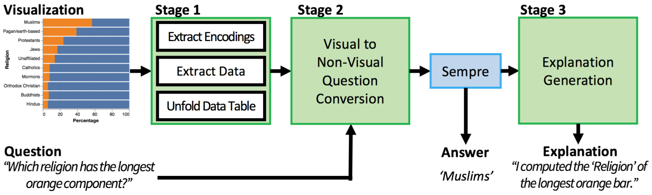
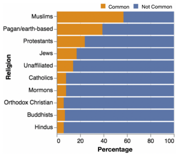


Figure 2. Our question answering pipeline for charts operates in three stages. In Stage 1, it extracts visual encodings and the data from the chart and then restructures the data table. In Stage 2, it transforms the input question, replacing any visual references to chart elements with non-visual references to data. Then, it passes the restructured data table and the transformed question to Sempre [43, 59], a state-of-the-art table question answering system which generates the text answer. Finally, in Stage 3, it generates a natuaral language explanation describing how the answer was generated from the chart.



```
"data": {"url": "data/kong/data/3.csv"},
"transform": [
    {"filter": "datum.year == 2000"},
    {"filter": "datum.question == 'Extremism'"}],
"mark": "bar",
"encoding": {
    "x": {"field": "Percentage", "type": "quantitative"},
    "y": {"field": "Religion", "type": "nominal"},
    "color": {
        "field": "Response", "type": "nominal",
        "scale": {
            "domain": ["Common", "Not common"],
            "range": ["#EE8426", "#5376A7"]}}}
```

Figure 3. Vega-Lite specification for the chart in Figure 1. This specification includes a block of data *"transforms"* (orange keyword text) that filter the data to specific years and questions. The *"mark"* (blue keyword) is specified as *'bar'*, and the visual *"encodings"* (pink keyword) for x-position, y-position, and color of the marks are given explicitly.

Religion	Response	Percentage
Muslims	Common	57
Muslims	Not common	43
Pagan/earth-based	Common	39
Pagan/earth-based	Not Common	61
⋮	⋮	⋮
Hindus	Common	6
Hindus	Not Common	94

(a) Flat data table

Religion	Common	Not common
Muslims	57	43
Pagan/earth-based	39	61
Protestants	24	76
Jews	17	83
⋮	⋮	⋮
Buddhists	7	93
Hindus	6	94

(b) Unfolded data table

Figure 4. (a) Data extracted from the chart in Figure 1 is initially a flat relational data table. Each row represents one mark in the chart. (b) We unfold the table by choosing the *'Response'* column as a pivot, turning each of its data values into column headers and then re-aligning the data in the other columns. In (a), each *'Religion'* and *'Response'* value appears multiple times but only once in (b) reducing the size of the table by almost a factor of two. Moreover, in (b), looking up a specific (*'Religion'*, *'Response'*) pair such as (*'Hindus'*, *'Not common'*) requires looking for the value at the intersection of the pair rather than searching through all the rows corresponding to Hindus as in (a).

to the attributes of visual marks (e.g. height of a bar mark, color of lines, etc.). We also extract the data itself from the input chart. In Stage 2, we use the extracted encodings to transform the input question, replacing all references to visual marks and their attributes with references to data fields and data values. This transformation converts a *visual question* into a purely *non-visual question*. Next, we input the unfolded table and the transformed, non-visual question into Sempre to generate the answer. Sempre converts the input natural language question into a logical query called a *lambda expression*, and then executes the query on the data table to generate the answer. Finally, in Stage 3, we convert the lambda expression from Sempre into a visual explanation for the answer, using template-based translation.

Stage 1: Extract Data Table and Encodings

A chart is typically constructed by encoding (or mapping) the data to some visual attributes (e.g. position, area, color) of graphical marks (e.g. circles, rectangles) [41]. Vega-Lite [50] is a chart specification language that explicitly describes how input data should be transformed (e.g. aggregating it, rescaling it) to make it suitable for visualization, and how the transformed data should be encoded using visual attributes of the marks (Figure 3).

In Stage 1 of our pipeline, we convert an input chart into a Vega-Lite specification and then extract encodings as well as the transformed data. Finally, we unfold the extracted data into a data table. We first describe how our extraction process works for a Vega-Lite chart and then explain how we convert other types of input charts into the Vega-Lite format.

Extraction from Vega-Lite Charts

Extract encodings. Given a Vega-Lite chart specification as in Figure 3, we can directly extract the encodings by looking for `encoding` keyword. In this example, the y-position attribute of a bar mark encodes a nominal data field named *'Religion'*, while the length attribute of the bar encodes a quantitative data field named *'Percentage'*. Similarly, the color attribute encodes a nominal data field named *'Response'* that takes the value 🟧 #EE8426 for the response *'Common'* or the value 🟦 #5376A7 for the response *'Not common'*.

Extract data. To extract the data from the chart, we first run the Vega-Lite interpreter and apply all data transformations in the chart specification. We then capture the transformed data from the Vega-Lite interpreter just before it is rendered into a chart. Specifically, we instrument the chart specification to write out the data immediately after the last transformation. The resulting data is in the form of a *flat* relational table where each row represents a single mark in the chart, or equivalently a single data tuple (Figure 4a).

Unfold data table. Table question answering systems like Sempre are trained using human readable tables which are often structured in an *unfolded* format in which a data tuple is comprised of a row header, a column header and the data value at their intersection (Figure 4b). Human readers typically prefer such unfolded tables to the corresponding flat relational tables because they are more compact and thereby reduce the cognitive effort required to retrieve information.

Sempre has been trained on a large set of tables from the WikiTableQuestions [43] dataset where manual inspection shows that many of the tables are unfolded. Therefore, we unfold our flat relational data tables into a form that is closer to Sempre's training data. Specifically, we implement Raman and Hellerstein's [46] `unfold` operation as follows. We first check that the extracted data table has a *pivot column* whose data values will be transformed into column headers. The pivot column must contain data values that repeat with the same frequency greater than one. In our example, the *'Religion'* column repeats each religion with a frequency of 2, while the *'Response'* column repeats each response with a frequency of 10 – each response appears once for each of the 10 religions in the dataset. We choose the column with the largest frequency as the pivot and re-align the data values in the other columns to form the unfolded table (Figure 4b). The resulting unfolded data table is passed as input to Sempre.

Converting Charts into Vega-Lite

While our extraction procedures are designed for charts specified using Vega-Lite, we can handle other forms of charts by converting them into the Vega-Lite format. For visualizations created using D3.js [2], we apply the D3 deconstructor of Harper and Agrawala [23, 24] to automatically convert them to Vega-Lite. The most prevalent representation of charts today is a bitmap. For such chart images, we first use ReVision [51] to extract the data and the marks and then manually add the visual encodings to convert the chart into a complete Vega-Lite specification. We leave it to future work to incorporate alternate methods for extracting data, marks and visual encodings [16, 44] from bitmaps of charts.

Stage 2: Visual to Non-Visual Question Conversion

In Stage 2, we transform an input question which may refer to visual aspects of the chart such as its marks and visual attributes, into a non-visual question that only refers to the data depicted in the chart. For example, consider the chart in Figure 1 and visual question Q2, *"Which religion has the longest orange component?"* Our goal is to convert this visual question into the corresponding non-visual question, *"Which religion has the most Percentage of Common Response data?"*

The visual version of the question uses the word *'component'* to refer to marks (bar segments) and the word *'orange'* to refer to a value (orange) of the visual attribute (color) of the marks. The word *'longest'* refers to performing an `argmax` operation on the visual attribute (length) over the marks, and we call such operations on visual attributes *visual operations*. Our approach identifies references to marks, visual attributes and visual operations in the question and then converts them to references to the data and operations on the data to build the the non-visual question in a sequence of 6 steps (Figure 5).

Step 1: Mark detection. The first step is to detect all words referring to graphical marks in the chart. Our approach is to check whether each word in the question appears in a list of *mark words* that we manually built in a one-time pre-process from our analysis of the additional set of visual questions we collected in the formative study (Figure 6 cyan). For instance, one may refer to a bar in a stacked bar chart as a *'component'*, *'portion'* or *'segment'* and we include these words in the list for bar marks. Thus for the question *"Which religion has the longest orange component?"*, we detect the word *'component'*, as referring to the bar marks in the stacked bar chart (Figure 5).

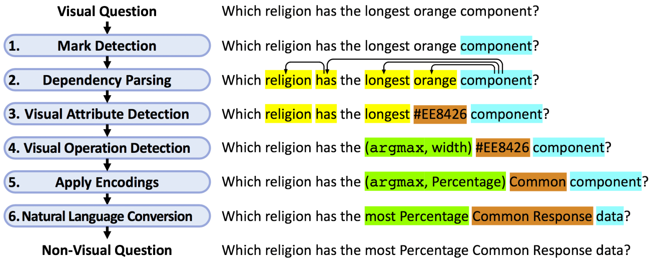
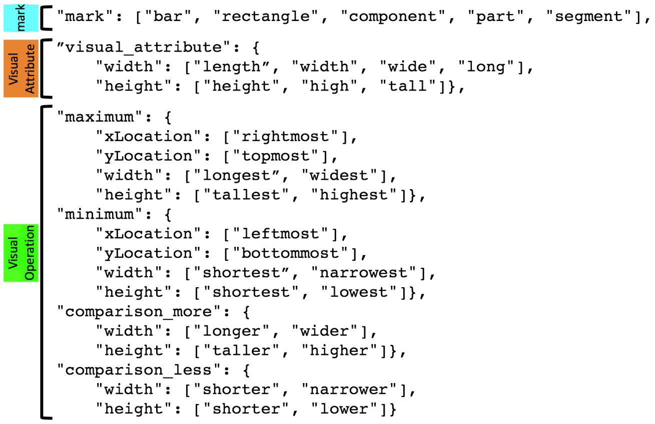

Visual Question
Which religion has the longest orange component?
1. Mark Detection
Which religion has the longest orange component?
2. Dependency Parsing
Which religion has the longest orange component?
3. Visual Attribute Detection
Which religion has the longest #EE8426 component?
4. Visual Operation Detection
Which religion has the (argmax, width) #EE8426 component?
5. Apply Encodings
Which religion has the (argmax, Percentage) Common component?
6. Natural Language Conversion
Which religion has the most Percentage Common Response data?
Non-Visual Question
Which religion has the most Percentage Common Response data?

Figure 5. Six steps used to convert the visual question (Q2 in Figure 1) into a non-visual question. The system detects *'component'* as a reference to the bar marks, it detects *'orange'* as a visual attribute word, and it detects *'longest'* as a visual operation word. It rewrites these words and outputs the rewritten non-visual question.


mark
Visual Attribute
Visual Operation

```
"mark": ["bar", "rectangle", "component", "part", "segment"],
”visual_attribute": {
    "width": ["length”, "width", "wide", "long"],
    "height": ["height", "high", "tall"]},
"maximum": {
    "xLocation": ["rightmost"],
    "yLocation": ["topmost"],
    "width": ["longest”, "widest"],
    "height": ["tallest", "highest"]},
"minimum": {
    "xLocation": ["leftmost"],
    "yLocation": ["bottommost"],
    "width": ["shortest”, "narrowest"],
    "height": ["shortest", "lowest"]},
"comparison_more": {
    "width": ["longer", "wider"],
    "height": ["taller", "higher"]},
"comparison_less": {
    "width": ["shorter", "narrower"],
    "height": ["shorter", "lower"]}
```

Figure 6. Word lists used in our pipeline for marks of type *'bar'*. The list of alternative words for referring to *'bar'* marks (cyan). The list of alternative words to refer to visual attributes of *'bar'* marks (orange). The list of visual operations and the alternative natural language word (e.g. rightmost, bottommost, wider, narrower) corresponding to each one (green) In this case the list also includes a visual attribute (e.g. xLocation, height) that the visual operation applies to. The operations are given at the top level and the visual attributes they operate on are given in the second level. The word lists for line charts are included in the Supplemental Materials.

Step 2: Dependency parsing. In the second step, we identify a set of words describing each graphical mark based on the grammatical structure of the question. We start by applying the Stanford CoreNLP dependency parser [37, 26] to obtain a parse tree that encodes phrase-level dependency structure. For instance, in the dependency tree for our example question (Figure 7), the word *'orange'* is an adjectival modifier `amod` for mark word *'component'*.

To obtain the words describing each mark, we find the tree node for each mark word in the question and traverse outwards following edges to its parents and children in breadth first order. We retain only the words corresponding to the following set of dependency labels: `acl`, `amod`, `compound`, `conj`, `dep`, `dobj`, `nmod`, and `nsubj`. We obtained this list by analyzing the complete set of dependency labels [4] along with our sample of additional visual questions collected in the formative study and noting that these labels best captured the visually descriptive words for each mark.

For our example (Figure 7), we traverse the tree starting at the mark word *'component'* and add the `amod` words *'longest'* and *'orange'* to the list of descriptive words, but we do not add the `det` word *'the'*. We would also traverse up the tree adding the parent word *'has'* with the relation `dobj` and then down through its children adding the `nsubj` word *'religion'*, but not the `det` word *'Which'*. Thus, for the mark word *'component'* we obtain the following set of descriptive words *'religion'*, *'has'*, *'longest'*, and *'orange'* as shown in Figure 5.

Questions often use color words to refer to marks without including a mark word. Thus, if the question contains a color word, we always add it to the list of descriptive words.

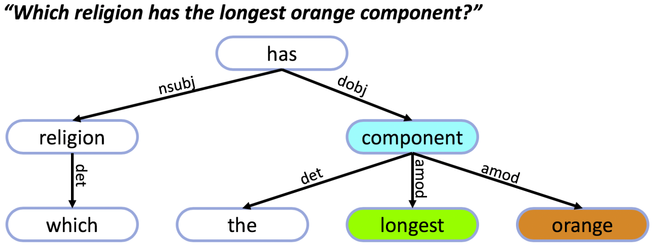


Figure 7. The dependency tree generated by the Stanford CoreNLP dependency parser for the question *"Which religion has the longest orange component?"*. Words comprising a noun phrase have the same parent noun in the tree and the tree provides a dependency relationship label for each edge (e.g. `amod` is an adjectival modifier, `det` is a determiner, `nsubj` is a nominal subject). In this case, the word for the visual attribute *'orange'* (orange) and the word for the visual operation *'longest'* (green) are *adjectival modifiers* (`amod`) of the mark word *'component'* (cyan)

Step 3: Visual attribute detection. We next identify all the visual attribute words in the list of descriptive words. Our approach is similar to the approach in step 1 for mark detection.

As a one-time pre-process, we built a list of attribute words for each mark type by analyzing our sample of example visual questions. In this analysis, we noticed that the same word can refer to a different visual attribute depending on the mark type (e.g., the word *'height'* refers to the *'length'* of a bar in a vertical bar chart whereas it refers to the *'y-position'* of a point in a line chart), so we created a separate list of visual attribute words for each mark type. The field `visual_attribute` in Figure 6 (labeled in orange) shows an example of the alternatives word list for visual attributes of bar marks.

We next filter the complete visual attributes list to just the ones that appear in the visual encodings we extracted from our chart in Stage 1. Then to identify which words in our descriptive words list refer to visual attributes, we iterate over each descriptive word and find the closest match in our filtered visual attributes list. Since there are lots of ways to describe visual attributes, we use a *word2vec*-based synonym finding approach to detect a match. Specifically, for each descriptive word and each filtered visual attribute word, we lookup the 300-dimensional word2vec vector generated by the pre-trained model of Mikolov et al. [39] trained on the Google News dataset [1]. We then compute the cosine similarity between their word2vec vectors and accept the best similarity match above a threshold τ (empirically set to 0.75).

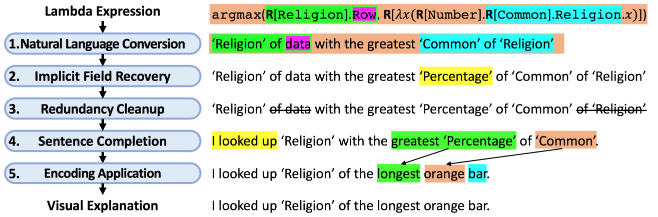
Questions sometimes contain descriptive words referring to a color (e.g. *'orange', 'red', 'blue'*). Such color words are generally ambiguous as the word *'red'* may refer to a range of different RGB values. Thus, whenever we encounter a descriptive color word, we first lookup the descriptive word in the text color names of the X11 color list [6] to obtain the corresponding RGB hex code. We then consider any encoding involving the *color* attribute, and examine all the RGB values the attribute takes within the chart. Finally, we replace the color word in the question with the RGB hex code that appears in the chart and is closest (in Euclidean RGB distance) to the X11 RGB hex code. In our example, the descriptive word *'orange'* yields the X11 hex code 🟧 #FFA500 and of the two colors 🟧 #EE8426 and 🟦 #5376A7 that appear in the chart (Figure 1), it is closest to the former. Therefore, we replace the word *'orange'* with 🟧 #EE8426 as shown in Figure 5.

Step 4: Visual operation detection. In step 4, we identify all the visual operation words in our remaining list of descriptive words. As in steps 1 and 3, we performed a one-time pre-process to build lists of alternative visual operation words (e.g. longest, narrowest, etc.). Each such visual operation word (e.g. tallest) implies performing an operation (e.g. `argmax`) on a specific visual attribute such as the *height* of a mark. Therefore, our visual operations word list maintains an (operation, attribute) pair for each visual operation word (Figure 6 green). To match a descriptive word to the visual operation words, we use the word2vec approach we used in step 3. In our example, we detect *'longest'* as a visual operation word and interpret it as the visual operation (`argmax`, width) as shown in Figure 5.

Note that we identify simple questions about the encoding (e.g. *"What is blue depicting?"*) by checking if the input question only refers to one visual feature or attribute of the chart and does not refer to a mark, data or visual operation. In such cases, we directly use the encodings we identified in Stage 1 to answer the question (e.g. *"Not Common"*), bypassing the rest of Stage 2.

Step 5: Apply encodings. In step 5, we use the encodings extracted in Stage 1 to replace the words corresponding to visual attributes and visual operations with words corresponding to data fields and data values. Specifically, we replace visual attribute words to the corresponding data field it encodes as given in the encoding. For specific visual attribute values like the orange color 🟧 #EE8426 we extracted in the step 3, we replace the attribute value with the corresponding data values – in this case the response *'Common'*. For visual operation words, we lookup the corresponding (operation, attribute) pair and replace the visual attribute with the corresponding data field based on the corresponding encoding. For example, given the visual operation word *'longest'*, we lookup the (`argmax`, width) pair, then find the encoding for the *width* attribute in the Vega-Lite specification and finally replace the attribute width with the corresponding data field *'Percentage'* from the encoding. Thus, we interpret the operation `argmax` as acting on the data field *'Percentage'* (Figure 5).

Step 6: Natural language conversion. In the final step, we convert our question into a non-visual natural language question suitable for input into Sempre, by rewriting words representing marks, visual attributes and visual operations using natural language equivalents. Because a mark represents a piece of data, we rewrite all mark words with the generic noun *'data'*. In Step 5, we converted the the visual attribute words into a corresponding data field or data value and we consider these as already in natural language. In our example, *'orange'* has already been converted into the data value *'Common'*. If as in this case the attribute word refers to a data value, we append the corresponding data field name to indicate the context in which the data value should be interpreted – in this case we append the data field name *'Response'* to the data value *'Common'*. Finally, if the attribute word is used as a noun, we add the word *'value'* to force the resulting conversion into a noun. For the visual operation words, we replace the opera-



argmax(**R**[Religion].Row, **R**[λx(**R**[Number].**R**[Common].Religion.x)])

'Religion' of data with the greatest 'Common' of 'Religion'

'Religion' of data with the greatest 'Percentage' of 'Common' of 'Religion'

'Religion' ~~of data~~ with the greatest 'Percentage' of 'Common' ~~of 'Religion'~~

I looked up 'Religion' with the greatest 'Percentage' of 'Common'.

I looked up 'Religion' of the longest orange bar.

I looked up 'Religion' of the longest orange bar.

Figure 8. Steps for generating an explanation from the lambda expression given by Sempre for the input question *"Which religion has the longest orange component?"* (Q2 in Figure 1). The system first converts the lambda expression generated by Sempre into a non-visual natural explanation. It then converts the non-visual explanation to a visual explanation by applying the visual encodings.

[Argmax]	`argmax(arg1,` **R**[λx(`arg2`.x)])	`arg1` with the greatest `arg2`
[Argmin]	`argmin(arg1,` **R**[λx(`arg2`.x)])	`arg1` with the smallest `arg2`
[Difference]	`-(arg1, arg2)`	difference between `arg1` and `arg2`
[Sum]	`sum(arg1)`	sum of `arg1`
[Count]	`count(arg1)`	number of `arg1`
[Lookup]	**R**`[property1].arg2`	`property1` of `arg2`
[Type]	**R**`[type1].arg2`	`arg2` (no-op)
[Row]	`Row`	data

Figure 9. Rules for converting operations in lambda expressions to natural language for some of the common operations. First column shows the name of the rule, the second column shows the labmda expression and the third column shows the corresponding natural language expression. We include more rules in the Supplemental Materials.

tion word pairs e.g. (`argmax`, *'Percentage'*) with the natural language equivalent of the operation while removing pair notation e.g. *'most Percentage'*. Thus, the input question *"Which religion has the longest orange component?"* is rewritten as *"Which religion has the most Percentage Common Response data?"* While the non-visual question is not completely fluent, together with our unfolded data table it contains enough information for Sempre to answer it correctly: *"Muslims"*.

Stage 3: Explanation Generation

In Stage 3, our pipeline generates a visual explanation describing how the answer was extracted from the chart's visual features. Our approach takes the logical query *lambda expression* Sempre builds to answer the question and uses template-based natural language generation to produce the explanation.

Consider the example question *"Which religion has the longest orange component?"* (Q2 in Figure 1). In Stage 2, we generate the corresponding non-visual question *"Which religion has the most Percentage Common Response data?"*. Sempre then converts this question into the lambda expression

$$\texttt{argmax}(\mathbf{R}[\texttt{Religion}].\texttt{Row},\\ \mathbf{R}[\lambda x[\mathbf{R}[\texttt{Number}].\mathbf{R}[\texttt{Common}].\texttt{Religion}.x]]),$$

which it executes on the unfolded table we generated in Stage 1 to produce the correct answer *'Muslims'*. Our goal in Stage 3 is to convert this lambda expression to the natural language visual explanation *"I computed the 'Religion' of the longest orange bar."* We use a 5 step pipeline to generate the explanation (Figure 8).

For questions about the encoding that we detected in step 4 of Stage 2 (e.g. *"What is blue depicting?"*), we directly generate the explanation using the template *"I looked up what [encoding] represents by looking at the [label on the x-axis / label on the y-axis / legend],"* based on whether the encoding is specified by the x-axis, the y-axis or the legend. We do not process such questions through the steps in this stage.

Step 1: Natural language conversion. As presented by Liang [35], lambda expressions include a limited set of operations and generation rules. Thus, we build a small set of rules to convert lambda expression to natural language (a subset of our rules is shown in Figure 9). For our example, our pipeline applies the `argmax`, `type`, `lookup`, and `row` rules to convert the input lambda expression to *"'Religion' of data with the greatest 'Common' of 'Religion'."*

Step 2: Implicit field recovery. Sometimes, a field name becomes implicit during the table unfolding in Stage 1, and we maintain the field name as an auxiliary annotation to the table. For instance, during the unfolding process in Figure 4, we keep the field name *'Percentage'* as auxiliary annotation on each of the cells in the *'Common'* and *'Not common'* columns. In this step we add this implicit annotation to the reference to the value *'Common'* of the pivoted field in the explanation, resulting in *"'Religion' of data with the greatest 'Percentage' of 'Common' of 'Religion'."*

Step 3: Redundancy Cleanup. Our pipline next removes any redundant information using a series of regex rules. In our explanation, we see that the information about *'Religion'* is repeated twice at the beginning and end of the expression. Moreover, *"'Religion' of data"* does not carry more information than just *'Religion'*. Both of these issues make the explanation difficult to understand. The cleanup step removes these extraneous words and yields *"'Religion' with the greatest 'Percentage' of 'Common'."* We include the specific regex rules in the Supplemental Materials.

Step 4: Sentence Completion. Next, we generate a non-visual explanation by adding the pronoun *'I'* and a verb that describes the last operation performed by the system. For the verb, we use *'looked up'* for lookup operations, *'counted'* for counting operations, and *'computed'* for all other operations. In our example, we add **"`I looked up`"** to the beginning of the explanation to complete a non-visual explanation.

Step 5: Encoding application. To make the non-visual explanations visual, we apply the visual encodings obtained from Stage 1. For references to values of fields that are encoded as colors, we convert them to color words directly. For references to data fields encoded as other visual features, we check the surronding words to see if there is an operation performed on the visual attribute, and convert it to a visual attribute word or a visual operation word using the word lists we used in Stage 2 (Figure 6). We add a mark word and position the converted visual words so that they modify the mark word. In our example, we convert the value *'Common'* to the color 'orange'. For the reference to the field *'Percentage'*, we use the neighboring word *'greatest'* and the visual encodings to recognize that this is an operation `argmax` on the visual attribute *width*, and use the visual operations word list to convert this to the visual operation word *'longest'*. Rearranging these words so that they modify the mark word *'bar'* yields *" I looked up 'Religion' of*

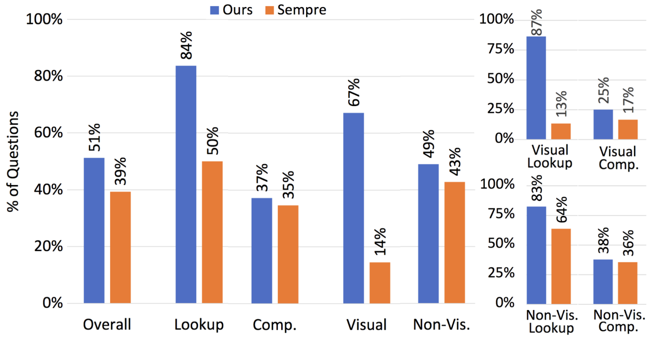


Figure 10. Accuracy of our pipeline (blue) compared to a baseline version of Sempre (orange) for questions of each type (visual/non-visual and lookup/compositional).

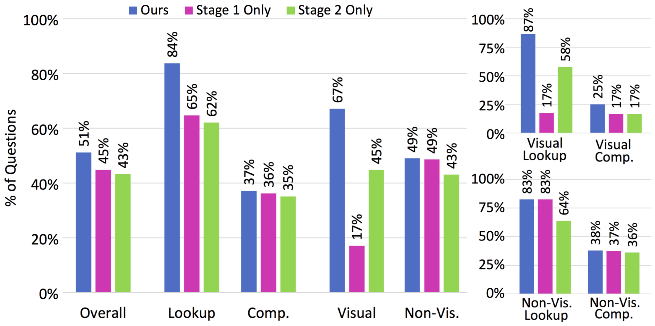


Figure 11. Accuracy of our complete pipeline (blue) compared to the pipeline with the data table unfolding of Stage 1 only (purple) and the question transformation of Stage 2 only (green).

the longest orange bar." Details about choice of color words and word rearrangements are in the Supplemental Materials.

RESULTS

As shown in Figure 10, we find that across all 629 questions in our corpus, our pipeline answers 51% correctly. As a baseline, we compare this result to using Sempre with the flat relational tables initially extracted in Stage 1 in place of the charts and find that it only answers 39% of the questions correctly. Our pipeline greatly outperforms Sempre on visual questions with improvements of 53% for all visual questions, 74% on visual lookup questions and 8% on visual compositional questions. We find that for even for non-visual questions, our system outperforms Sempre, by 6% overall, 19% on non-visual lookup questions and 2% on non-visual compositional questions.

Figure 11 compares the accuracy of our complete pipeline to a pipeline in which we only retain Stage 1 (and eliminate Stage 2—visual to non-visual conversion) and to a pipeline in which we only retain Stage 2 (more specifically, we include data and encoding extraction from Stage 1 but eliminate data table unfolding). Although both stages contribute significantly to the overall success of our pipeline, we see a major improvement in answering visual questions from Stage 2, which is not surprising as Stage 2 is responsible for converting visual questions into the non-visual form necessary for Sempre.

Figure 12 shows a variety of charts and questions with answers and explanations generated by our pipeline, as well as the answers generated by the baseline version of Sempre. We see that our system generates correct answers and explanations for many questions that Sempre cannot answer correctly. In particular, Stage 2 of our pipeline handles visual features and allows our pipeline to correctly answers visual questions (Q5, Q7, Q9, Q11, Q13, Q14). It even correctly answers non-visual questions both lookup (Q1, Q3, Q17) and compositional (Q2, Q4, Q8, Q10, Q19). However, our pipeline sometimes outputs a wrong answer for a question Sempre gets correct, as in Q18. In this case the error is due to a change in table structure from table unfolding in Stage 1 of our pipeline.

Explanation	Transparency	Trust	Usefulness	Accuracy(%)	Time(s)
None	1.3 (±0.8)	3.0 (±1.1)	-	87.5	26.2 (±28.3)
Human	3.3 (±0.8)	**3.3** (±0.9)	3.4 (±1.5)	91.3	26.0 (±27.5)
Non-Visual	3.4 (±1.3)	3.1 (±0.9)	3.6 (±1.4)	95.0	**23.7** (±21.2)
Visual	**3.9** (±1.1)	**3.3** (±0.9)	**3.7** (±1.5)	**98.8**	26.7 (±30.5)

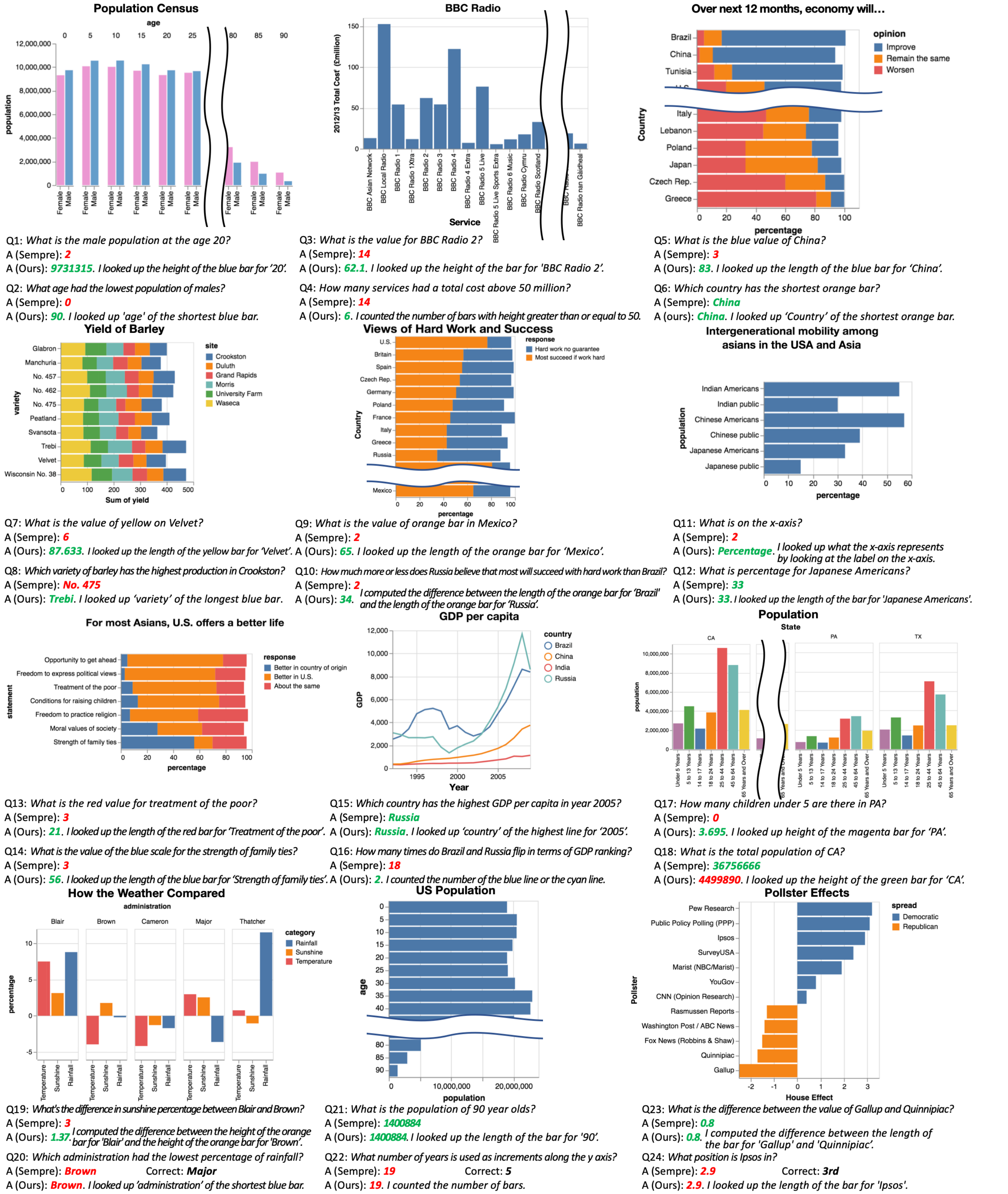
Table 2. Results from the user study (each result is represented as $avg(\pm stdev)$**). We see that for most of the measures, the visual explanations generated by our system achieves the best.**

Nevertheless, analyzing the wrong answers produced by our system, we find that 92% are due to Sempre, and 5% are due to incorrect conversions of visual questions. The remaining 4% are because of changes in the table structure from table unfolding. Further analyzing the Sempre errors, 12% are caused by Sempre not including the operation involved in the question. For example, Sempre does not include operations with binary output, making it unable to answer Y/N questions, which accounts for 1.5% of all the questions. We refer to the analysis in the original papers [59, 43] for more details about errors by Sempre.

Our system gets the correct answer for Q16, but from the explanation, we see that it counted the number of lines corresponding to the countries that appeared in the question (i.e. Brazil and Russia) instead of counting the number of flips in the GDP ranking of the two countries; it accidentally got the answer correct. On the other hand, since Sempre does not give explanations, it is unclear how many of the correct answers it gives are obtained through an incorrect process because the model is opaque. Even for questions our system gets wrong (Q18, Q20, Q22, Q24), we see that our system transparently explains how it arrived at the wrong answer.

USER STUDY

To see how the visual explanations generated by our pipeline do on the measures of transparency, trust, and usefulness, we conducted a user study with four different conditions: (1) the *no-explanation* condition in which we only show the answer to a question, (2) the *human explanation* condition in which we show the answers and explanations generated by humans from our formative study, (3) the *non-visual explanation* condition in which we show the answers and the non-visual explanations generated by our pipeline (at the end of step 4 of Stage 3), and (4) the *visual-explanation* condition in which we show the answers and visual explanations generated by our pipeline. We consider three hypotheses:
H1: Users will find the visual explanation condition more transparent and trustworthy than the no-explanation condition.
H2: Users will find the visual explanation condition better than or at least as good as human explanation condition based on transparency, trust, and usefulness.
H3: Users will find the visual explanation condition better



Q1: *What is the male population at the age 20?*
A (Sempre): ***2***
A (Ours): ***9731315***. *I looked up the height of the blue bar for '20'.*

Q2: *What age had the lowest population of males?*
A (Sempre): ***0***
A (Ours): ***90***. *I looked up 'age' of the shortest blue bar.*

Q3: *What is the value for BBC Radio 2?*
A (Sempre): ***14***
A (Ours): ***62.1***. *I looked up the height of the bar for 'BBC Radio 2'.*

Q4: *How many services had a total cost above 50 million?*
A (Sempre): ***14***
A (Ours): ***6***. *I counted the number of bars with height greater than or equal to 50.*

Q5: *What is the blue value of China?*
A (Sempre): ***3***
A (Ours): ***83***. *I looked up the length of the blue bar for 'China'.*

Q6: *Which country has the shortest orange bar?*
A (Sempre): ***China***
A (ours): ***China***. *I looked up 'Country' of the shortest orange bar.*

Q7: *What is the value of yellow on Velvet?*
A (Sempre): ***6***
A (Ours): ***87.633***. *I looked up the length of the yellow bar for 'Velvet'.*

Q8: *Which variety of barley has the highest production in Crookston?*
A (Sempre): ***No. 475***
A (Ours): ***Trebi***. *I looked up 'variety' of the longest blue bar.*

Q9: *What is the value of orange bar in Mexico?*
A (Sempre): ***2***
A (Ours): ***65***. *I looked up the length of the orange bar for 'Mexico'.*

Q10: *How much more or less does Russia believe that most will succeed with hard work than Brazil?*
A (Sempre): ***2***
A (Ours): ***34***. *I computed the difference between the length of the orange bar for 'Brazil' and the length of the orange bar for 'Russia'.*

Q11: *What is on the x-axis?*
A (Sempre): ***2***
A (Ours): ***Percentage***. *I looked up what the x-axis represents by looking at the label on the x-axis.*

Q12: *What is percentage for Japanese Americans?*
A (Sempre): ***33***
A (Ours): ***33***. *I looked up the length of the bar for 'Japanese Americans'.*

Q13: *What is the red value for treatment of the poor?*
A (Sempre): ***3***
A (Ours): ***21***. *I looked up the length of the red bar for 'Treatment of the poor'.*

Q14: *What is the value of the blue scale for the strength of family ties?*
A (Sempre): ***3***
A (Ours): ***56***. *I looked up the length of the blue bar for 'Strength of family ties'.*

Q15: *Which country has the highest GDP per capita in year 2005?*
A (Sempre): ***Russia***
A (Ours): ***Russia***. *I looked up 'country' of the highest line for '2005'.*

Q16: *How many times do Brazil and Russia flip in terms of GDP ranking?*
A (Sempre): ***18***
A (Ours): ***2***. *I counted the number of the blue line or the cyan line.*

Q17: *How many children under 5 are there in PA?*
A (Sempre): ***0***
A (Ours): ***3.695***. *I looked up height of the magenta bar for 'PA'.*

Q18: *What is the total population of CA?*
A (Sempre): ***36756666***
A (Ours): ***4499890***. *I looked up the height of the green bar for 'CA'.*

Q19: *What's the difference in sunshine percentage between Blair and Brown?*
A (Sempre): ***3***
A (Ours): ***1.37***. *I computed the difference between the height of the orange bar for 'Blair' and the height of the orange bar for 'Brown'.*

Q20: *Which administration had the lowest percentage of rainfall?*
A (Sempre): ***Brown*** Correct: ***Major***
A (Ours): ***Brown***. *I looked up 'administration' of the shortest blue bar.*

Q21: *What is the population of 90 year olds?*
A (Sempre): ***1400884***
A (Ours): ***1400884***. *I looked up the length of the bar for '90'.*

Q22: *What number of years is used as increments along the y axis?*
A (Sempre): ***19*** Correct: ***5***
A (Ours): ***19***. *I counted the number of bars.*

Q23: *What is the difference between the value of Gallup and Quinnipiac?*
A (Sempre): ***0.8***
A (Ours): ***0.8***. *I computed the difference between the length of the bar for 'Gallup' and 'Quinnipiac'.*

Q24: *What position is Ipsos in?*
A (Sempre): ***2.9*** Correct: ***3rd***
A (Ours): ***2.9***. *I looked up the length of the bar for 'Ipsos'.*

Figure 12. Sample questions from our corpus with answers generated by our pipeline as well as a baseline version of Sempre. Answers in green are correct and answers in red are incorrect. If neither of pipeline generated a correct answer, we also report the correct answer as in Q20, Q22, and Q24. We encourage readers to zoom in to the figure to read the text.

than the non-visual explanation condition based on transparency, trust, and usefulness.

Study Design

We designed a within-subjects study with sixteen participants, all fluent in English. To set up the study we gathered 20 unique charts-question pairs from our corpus, and divided them into four groups of five. We counterbalanced the mapping between the four conditions and the four chart-question groups. We then ran the study in two stages. In the first stage, we randomly shuffled the questions. Along with the chart and the question, we showed the participants answers and explanations (if any) of the condition the question was mapped to. For each question, the participants first determined whether the presented answer was correct, and then rated the usefulness of the explanation on a 5-point Likert scale. We timed how long it took them to determine correctness. In the second stage, we showed each group of questions in a counterbalanced order and asked the participants to rate the transparency and the trustworthiness of the condition on a 5-point Likert scale. Afterwards, we collected free form responses about what the participants considered relevant to the transparency, trustworthiness, and usefulness. The study took about 30 minutes and each participant received a $15.00 Amazon gift card.

Results and Discussion

Assessing H1. Figure 2 shows the results of the study. We find that the visual explanations generated by our pipeline significantly increased the transparency of the pipeline compared to the no-explanation condition (Mann-Whitney U = 245.5, $p < 0.001$). Trust towards the visual explanation condition was higher than the no-explanation condition, but the difference was not significant (U = 149.0, $p = 0.21$).

Assessing H2. Participants also found the visual explanations generated by our pipeline significantly more transparent than the human-generated explanations (U = 178.0, $p < 0.05$). We hypothesize that this result is due to the systematic way our pipeline generates the explanations, as one participant put it, *"I like that in some system the explanation is more consistent than others. It guarantees me that it will provide certain information."* Finally, we saw that the trust towards visual explanations generated by our pipeline is very close to that towards human-generated explanations (U = 130.5, $p = 0.46$).

Assessing H3. When we compare between the visual explanations to non-visual explanations generated by our pipeline, we find that the measures of transparency, trust, and usefulness are all higher for the visual explanations, but none of the improvements are significant (U = 153.5, $p = 0.16$; U = 143.5, $p = 0.27$; U = 3356.5, $p = 0.29$, respectively).

In the free form response, most participants (12 of 16) reported explanations as relevant to transparency (e.g., *"I appreciated being able to see from the explanations what caused the system to make errors. Providing no explanation at all made the system seem like a complete black box."*). For trust, participants reported the accuracy of the answer (10 of 16) and whether the explanation matches the answer (4 of 16) as relevant. Finally, for usefulness, participants reported that explanations are more useful if they refer to visual features (7 of 16).

In sum, we find the visual explanations generated by our system are significantly more transparent than human-generated explanations and are comparable in usefulness and trust.

Accuracy and Time. We also measured the accuracy and speed with which participants could confirm the correctness of answers with and without explanations. While these initial measurements show improvements in accuracy for and speed when people have access to explanations, the relatively small differences combined with large variance in timing, suggest that further study is needed to understand the causes of these improvements. We provide more details about these measurements in the Supplemental Materials.

LIMITATIONS AND FUTURE WORK

Although our question answering system for charts and graphs provides good accuracy, there are several limitations that we would like to lift in future work.

Handle additional types of questions. Although our pipeline achieves reasonable accuracy, there is room for improvement. For example, people refer to visual features of a chart using a variety of words. Our rule-based approach sometimes fails to detect synonyms for these features. A supervised method that learns to detect such visual references and convert visual questions to non-visual questions could provide a more generalizable model. We have also found classes of questions that our system cannot handle, some because Sempre [59] cannot handle them (e.g., yes/no questions), and some compositional questions about visual encodings (e.g., Q22 of Figure 12). We hope to extend our pipeline to handle such questions.

Improving explanations. While our template-based approach generates explanations that can convey how our pipeline obtained an answer, the resulting explanation may lack fluency and offer little variations in style. Applying data-driven neural models for natural language generation [38] may help address such limitations. Moreover, as Kong et al. [31] and Kim et al. [30] have suggested, highlighting parts of the charts relevant to the explanations might also improve their effectiveness.

CONCLUSIONS

We have presented an automatic pipeline for answering questions about charts and generating visual explanations. In a formative study, we find that people regularly ask visual questions and that visual explanations are both common and effective. Our automatic question-answering pipeline achieves an overall accuracy of 51% on a corpus of real-world chart with human-generated questions. Finally, user study confirms that our system is significantly more transparent than the answers and explanations generated by humans, and that it is on par with the human-generated answers and explanations for trust and usefulness. Our code and data are available at: `https://github.com/dhkim16/VisQA-release`

ACKNOWLEDGMENTS

The authors thank Panupong Pasupat for advice during pipeline development and Sean Liu for help with data annotation. This work is supported by an Allen Distinguished Investigator award and by NSF award III-1714647. Dae Hyun Kim is partly supported by a Samsung Scholarship.

REFERENCES

[1] 2018. Word embedding trained on Google News. (2018). Retrieved April 02, 2018 from https://code.google.com/archive/p/word2vec/.

[2] 2019. D3 JavaScript Library. (2019). Retrieved March 20, 2019 from https://d3js.org.

[3] 2019. Pew Research. (2019). Retrieved March 20, 2019 from http://www.pewresearch.org/.

[4] 2019. Universal dependencies. (2019). Retrieved March 20, 2019 from https://universaldependencies.org/.

[5] 2019. Vega-Lite Example Gallery. (2019). Retrieved March 30, 2019 from https://vega.github.io/vega-lite/examples/.

[6] 2019. X11 color names. (2019). Retrieved March 20, 2019 from https://en.wikipedia.org/wiki/X11_color_names.

[7] Amina Adadi and Mohammed Berrada. 2018. Peeking inside the black-box: A survey on Explainable Artificial Intelligence (XAI). *IEEE Access* 6 (2018), 52138–52160.

[8] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. 2018. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*. 9505–9515.

[9] Rishabh Agarwal, Chen Liang, Dale Schuurmans, and Mohammad Norouzi. 2019. Learning to Generalize from Sparse and Underspecified Rewards. *arXiv preprint arXiv:1902.07198* (2019).

[10] 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*. 2425–2433.

[11] Sriram Karthik Badam, Zhicheng Liu, and Niklas Elmqvist. 2019. Elastic Documents: Coupling Text and Tables through Contextual Visualizations for Enhanced Document Reading. *IEEE transactions on visualization and computer graphics* 25, 1 (2019), 661–671.

[12] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. 2017. Interpreting blackbox models via model extraction. *arXiv preprint arXiv:1705.08504* (2017).

[13] Leilani Battle, Peitong Duan, Zachery Miranda, Dana Mukusheva, Remco Chang, and Michael Stonebraker. 2018. Beagle: Automated Extraction and Interpretation of Visualizations from the Web. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 594, 8 pages. DOI: http://dx.doi.org/10.1145/3173574.3174168

[14] Ritwick Chaudhry, Sumit Shekhar, Utkarsh Gupta, Pranav Maneriker, Prann Bansal, and Ajay Joshi. 2019. LEAF-QA: Locate, Encode & Attend for Figure Question Answering. *arXiv preprint arXiv:1907.12861* (2019).

[15] Charles Chen, Ruiyi Zhang, Eunyee Koh, Sungchul Kim, Scott Cohen, Tong Yu, Ryan Rossi, and Razvan Bunescu. 2019. Figure Captioning with Reasoning and Sequence-Level Training. *arXiv preprint arXiv:1906.02850* (2019).

[16] Jinho Choi, Sanghun Jung, Deok Gun Park, Jaegul Choo, and Niklas Elmqvist. 2019. Visualizing for the Non-Visual: Enabling the Visually Impaired to Use Visualization. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 249–260.

[17] Kenneth Cox, Rebecca E Grinter, Stacie L Hibino, Lalita Jategaonkar Jagadeesan, and David Mantilla. 2001. A multi-modal natural language interface to an information visualization environment. *International Journal of Speech Technology* 4, 3-4 (2001), 297–314.

[18] Kedar Dhamdhere, Kevin S McCurley, Ralfi Nahmias, Mukund Sundararajan, and Qiqi Yan. 2017. Analyza: Exploring data with conversation. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces*. ACM, 493–504.

[19] Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608* (2017).

[20] Nicholas Frosst and Geoffrey Hinton. 2017. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784* (2017).

[21] Jianfeng Gao, Michel Galley, and Lihong Li. 2018. Neural approaches to conversational AI. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 1371–1374.

[22] Tong Gao, Mira Dontcheva, Eytan Adar, Zhicheng Liu, and Karrie G Karahalios. 2015. Datatone: Managing ambiguity in natural language interfaces for data visualization. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. ACM, 489–500.

[23] Jonathan Harper and Maneesh Agrawala. 2014. Deconstructing and restyling D3 visualizations. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. ACM, 253–262.

[24] Jonathan Harper and Maneesh Agrawala. 2017. Converting Basic D3 Charts into Reusable Style Templates. *IEEE transactions on visualization and computer graphics* (2017).

[25] Enamul Hoque, Vidya Setlur, Melanie Tory, and Isaac Dykeman. 2018. Applying Pragmatics Principles for Interaction with Visual Analytics. *IEEE transactions on visualization and computer graphics* 24, 1 (2018), 309–318.

[26] Daniel Jurafsky and James Martin. 2019. Dependency Parsing. *Speech and Language Processing* (2019). Retrieved March 20, 2019 from https://web.stanford.edu/~jurafsky/slp3/13.pdf.

[27] 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*. 5648–5656.

[28] Kushal Kafle, Robik Shrestha, Brian Price, Scott Cohen, and Christopher Kanan. 2019. Answering Questions about Data Visualizations using Efficient Bimodal Fusion. *arXiv preprint arXiv:1908.01801* (2019).

[29] 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* (2017).

[30] Dae Hyun Kim, Enamul Hoque, Juho Kim, and Maneesh Agrawala. 2018. Facilitating Document Reading by Linking Text and Tables. In *The 31st Annual ACM Symposium on User Interface Software and Technology*. ACM, 423–434.

[31] Nicholas Kong, Marti A Hearst, and Maneesh Agrawala. 2014. Extracting references between text and charts via crowdsourcing. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, 31–40.

[32] Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. 2017. Neural semantic parsing with type constraints for semi-structured tables. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 1516–1526.

[33] Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155* (2016).

[34] Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc Le, and Ni Lao. 2018. Memory augmented policy optimization for program synthesis with generalization. *arXiv preprint arXiv:1807.02322* (2018).

[35] P. Liang. 2013. Lambda Dependency-Based Compositional Semantics. *arXiv preprint arXiv:1309.4408* (2013).

[36] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical question-image co-attention for visual question answering. In *Advances In Neural Information Processing Systems*. 289–297.

[37] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*. 55–60.

[38] Hongyuan Mei, Mohit Bansal, and Matthew R Walter. 2016. What to talk about and how? Selective Generation using LSTMs with Coarse-to-Fine Alignment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 720–730.

[39] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[40] Vibhu O Mittal, Giuseppe Carenini, Johanna D Moore, and Steven Roth. 1998. Describing complex charts in natural language: A caption generation system. *Computational Linguistics* 24, 3 (1998), 431–467.

[41] Tamara Munzner. 2014. *Visualization Analysis and Design*. CRC Press.

[42] Menaka Narayanan, Emily Chen, Jeffrey He, Been Kim, Sam Gershman, and Finale Doshi-Velez. 2018. How do humans understand explanations from machine learning systems? an evaluation of the human-interpretability of explanation. *arXiv preprint arXiv:1802.00682* (2018).

[43] Panupong Pasupat and Percy Liang. 2015. 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)*, Vol. 1. 1470–1480.

[44] Jorge Poco and Jeffrey Heer. 2017. Reverse-Engineering Visualizations: Recovering Visual Encodings from Chart Images. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 353–363.

[45] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250* (2016).

[46] Vijayshankar Raman and Joseph M Hellerstein. 2001. Potter's wheel: An interactive data cleaning system. In *VLDB*, Vol. 1. 381–390.

[47] Revanth Reddy, Rahul Ramesh, Ameet Deshpande, and Mitesh M Khapra. 2019. FigureNet: A Deep Learning model for Question-Answering on Scientific Plots. In *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[48] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 1135–1144.

[49] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-precision model-agnostic explanations. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[50] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. 2017. Vega-lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 341–350.

[51] 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*. ACM, 393–402.

[52] Vidya Setlur, Sarah E Battersby, Melanie Tory, Rich Gossweiler, and Angel X Chang. 2016. Eviza: A natural language interface for visual analysis. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 365–377.

[53] Vidya Setlur and Melanie Tory. 2017. Exploring Synergies between Visual Analytical Flow and Language Pragmatics. (2017).

[54] Vidya Setlur, Melanie Tory, and Alex Djalali. 2019. Inferencing Underspecified Natural Language Utterances in Visual Analysis. In *Proceedings of the 24th International Conference on Intelligent User Interfaces (IUI '19)*. 40–51.

[55] Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. 2017. Reasonet: Learning to stop reading in machine comprehension. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1047–1055.

[56] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).

[57] Yiwen Sun, Jason Leigh, Andrew Johnson, and Sangyoon Lee. 2010. Articulate: A semi-automated model for translating natural language queries into meaningful visualizations. In *International Symposium on Smart Graphics*. Springer, 184–195.

[58] Caiming Xiong, Victor Zhong, and Richard Socher. 2016. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604* (2016).

[59] Yuchen Zhang, Panupong Pasupat, and Percy Liang. 2017. Macro Grammars and Holistic Triggering for Efficient Semantic Parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1214–1223.

[60] Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103* (2017).