AI4VIS: Survey on Artificial Intelligence Approaches for Data Visualization

Aoyu Wu ⓘ, Yun Wang ⓘ, Xinhuan Shu ⓘ, Dominik Moritz ⓘ, Weiwei Cui, Haidong Zhang, Dongmei Zhang, and Huamin Qu

Abstract—Visualizations themselves have become a data format. Akin to other data formats such as text and images, visualizations are increasingly created, stored, shared, and (re-)used with artificial intelligence (AI) techniques. In this survey, we probe the underlying vision of formalizing visualizations as an emerging data format and review the recent advance in applying AI techniques to visualization data (AI4VIS). We define visualization data as the digital representations of visualizations in computers and focus on data visualization (e.g., charts and infographics). We build our survey upon a corpus spanning ten different fields in computer science with an eye toward identifying important common interests. Our resulting taxonomy is organized around WHAT is visualization data and its representation, WHY and HOW to apply AI to visualization data. We highlight a set of common tasks that researchers apply to the visualization data and present a detailed discussion of AI approaches developed to accomplish those tasks. Drawing upon our literature review, we discuss several important research questions surrounding the management and exploitation of visualization data, as well as the role of AI in support of those processes. We make the list of surveyed papers and related material available online at.

Index Terms—Survey, data visualization, artificial intelligence, data format, machine learning

✦

1 INTRODUCTION

DATA visualizations use visual representations of abstract data to amplify human cognition. Researchers traditionally investigate visualizations as artifacts created for people. This paper revisits this traditional perspective in line with the growing research interest in applying artificial intelligence (AI) to visualizations. Similar to common data formats like text and images, visualizations are increasingly created, shared, collected, and reused with the power of AI. Thus, we see that visualizations are becoming a new data format processed by AI. For instance, this trend was evident at the 2020 IEEE Visualization Conference, where multiple techniques were proposed for automating the creation of visualizations [1], [2], [3], [4], [5], retargeting visualizations [6], [7], and analyzing visualization ensembles [8], [9]. In light of this trend, new concepts and research problems are emerging, raising the need to organize existing literature and clarify the research landscape.

This survey describes the research vision of formalizing visualizations as an emerging data format and reviews recent advances in developing AI approaches for visualization

data (AI4VIS). We define *visualization data* as the digital representations of visualizations in computers and focus on data visualization. Different from common formats, visualization data contains multimodal information such as visual encodings, encoded data, text, and images. Those characteristics pose new challenges in developing tailored AI approaches (e.g., how to represent visualization data).

Nevertheless, AI is a broad notion that has been studied in different areas. Those areas have different motivations and research questions about applying AI to visualization data, the proposed techniques, and the content format used to represent visualizations. For instance, the *Web and Information Retrieval* community advances the technology for searching visualizations (e.g., [10], [11]), while research in *Computer Vision* recently studies visual question answering in charts (e.g., [12], [13]). Therefore, a comprehensive understanding of AI4VIS research requires a foundation in rich literature from diverse disciplines.

We construct a literature corpus by a relation-search approach [14], i.e., graph traversal over the citation and reference networks. This approach allows us to collect 98 publications from 10 research communities – publications from the visualization community account for roughly one-third. While not being exhaustive, our corpus provides sufficient research instances to synthesize relevant work that contributes to the understandings of important common research questions and techniques. As shown in Fig. 1, we organize and categorize AI4VIS research following a well-established what-why-how viewpoint [15]:

- *What* is visualization data. We formalize the concept of visualization data by providing an overview of its content formats as well as the representations (Fig. 1(A)).
- *Why* apply AI to visualization data. We identify three common goals for AI4VIS research, namely

• *Aoyu Wu, Xinhuan Shu, and Huamin Qu are with the Hong Kong University of Science and Technology, Hong Kong. E-mail: {awuac, xinhuan. shu, huamin}@cse.ust.hk.*
• *Yun Wang, Weiwei Cui, Haidong Zhang, and Dongmei Zhang are with the Microsoft Research Asia, Beijing 100080, China. E-mail: {wangyun, weiwei.cui, haidong.zhang, dongmeiz}@microsoft.com.*
• *Dominik Moritz is with Carnegie Mellon University, Pittsburgh, PA 15213 USA, and also with Apple, Cupertino, CA 95014 USA. E-mail: domoritz@cmu.edu.*
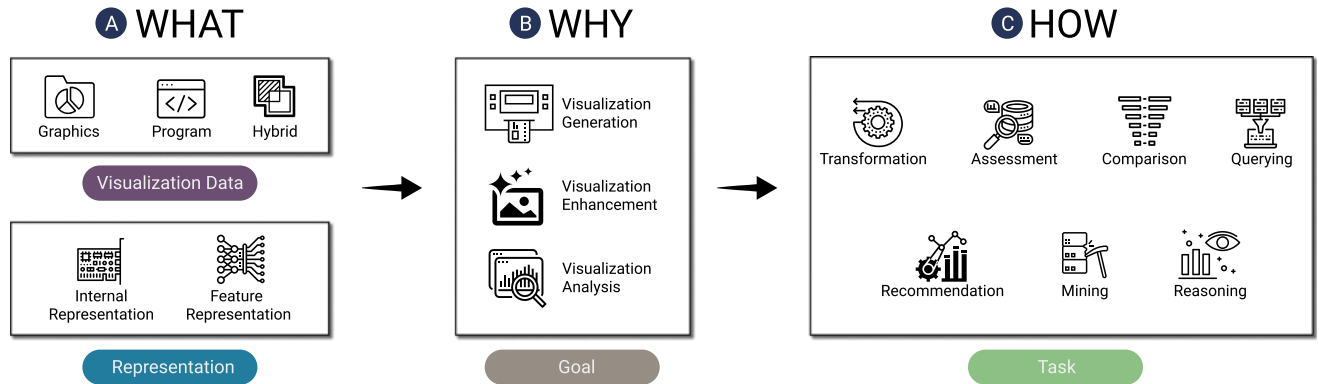
Fig. 1. We categorize the surveyed papers from three aspects: (A) What is the visualization data and its representation; (B) Goals for why apply AI to visualization data; and (C) How to apply AI to visualization data, where we outline seven tasks and separately discuss corresponding AI approaches.

visualization generation, enhancement, and analysis (Fig. 1(B)). We classify those goals into subcategories to provide a comprehensive response to ongoing discussions like "why should we teach machines to read charts made for humans" [16].

- *How* to apply AI to visualization data. Most importantly, we contribute a task abstraction regarding how to apply AI to visualization data (Fig. 1(C)). The task abstraction is critical since it allows for domain-agnostic and consistent descriptions of research questions across different disciplines. Besides, it facilitates an organized discussion reconciling distinct visualization paper types [17], i.e., a *technique* paper focuses on a single task, while a *system* paper might accomplish multiple tasks. In total, we note seven common tasks and discuss the AI approaches for each task separately.

Drawing upon the discussion, we outline research opportunities. We make the list of surveyed papers with related material available online at ai4vis.github.io. We hope that our survey will stimulate new theories, problems, techniques, and applications in this growing research area.

2 RELATED SURVEYS

Our ability to collect data has significantly exceeded our ability to analyze it, contributing to the emergence of AI approaches that automate the processes. Recently, there are active ongoing discussions about how visualization research could be interwoven with artificial intelligence (AI) [18], [19]. However, AI has been defined and operationalized as a broad notion. To clarify our discussion, our specific prospect is to formalize visualizations as an emerging format of data. We see visualizations different from existing types such as images and texts, thereby raising many research questions regarding how AI facilitates the manipulation and analysis of visualizations.

Several surveys review techniques for automating the creation of visualizations. Saket *et al.* [20] discussed the prospect of learning visualization design and classified automated visualization design systems into knowledge-based (i.e., rule-based), data-driven (i.e., machine-learning), and hybrid approaches. This classification was systematically reviewed in a recent survey about visualization and infographics recommendation [21]. Besides, Qin *et al.* [22]

drew on research in the database community to survey what makes data visualization more efficient and effective. In addition to automated creation and recommendation, Davila *et al.* [23] reviewed research over the past eight years to formalize *chart mining*, defined as "the process of automatic detection, extraction and analysis of charts to reproduce the tabular data that was originally used to create them". Wang *et al.* [24] recently surveyed machine learning models applied to visualizations. Different from them, our objective is to provide a comprehensive review of AI4VIS research, that is, a scaffold for emerging research problems to be formulated and understood (e.g., automatic assessment and summarization of visualizations).

3 METHODOLOGY

In this section, we describe the scope of our literature survey, the search methodology and corpus, as well as our analysis method.

3.1 Definition and Scope

This survey focuses on AI approaches applied to visualization data. We define visualization data as the digital representations of visualizations in computers. Thus, we include existing work that contributes AI techniques or systems that primarily focuses on *inputting or outputting visualization data*. However, due to the wide scope of visualizations and related research, we constrict the scope of visualizations for manageability.

Excluding Scientific Visualizations. We cover literature related to information visualization and visual analytics, particularly for which visualizations are represented as charts and infographics. This restriction excludes literature that primarily focuses on scientific visualizations. Scientific visualizations represent scientific data such as flows and volumes, which are typically designed with a strong inherent reference to space and time [25]. We exclude them due to their heterogeneous nature that yields different research challenges and interests [15].

Excluding Research Specific to a Chart Type. The research problems and proposed techniques could vary among different chart types. For instance, a conference (i.e., International Symposium on Graph Drawing) in the graph visualization community focuses on the graph layout problem, whereas their problems might not apply to other

visualizations like histograms. Those chart-specific studies are large-scale and therefore might not be covered in a single survey, e.g., Behrisch *et al.* [26] devoted a survey to discussions about assessment metrics for different charts. Different from them, our goal is to identify common research problems (i.e., assessment, recommending) irrespective of the chart types. Nevertheless, we note a few chart-type-specific studies and discuss how they might fall into our taxonomy in Section 8.

Excluding Human Interaction Data. Finally, we emphasize the central role of visualization data in the surveyed AI approaches. In other words, we do not consider work whose primary goal is to collect and analyze human interaction data when creating or using visualizations. Therefore, we exclude provenance data [15] and data collected for natural language interfaces [27].

3.2 Method and Corpus

To establish the corpus of papers we discussed in this survey, we apply a relation-search method [14] to traverse the literature. Our method starts with a linear scan of full papers published at the 2020 IEEE Visualization Conference to collect the starting points. This initial set of papers has nine papers [1], [2], [3], [4], [5], [6], [7], [8], [9]. We further augment these starting points with papers covered in related surveys [20], [21], [22], [23]. If a paper is selected to be included, we traverse both its references and citations. We search for papers breadth-first in an attempt to avoid over-focus on one particular line of research.

Our relation-search method results in an interdisciplinary corpus consisting of 98 papers from 10 research areas. We decide the research area of each paper according to the classification of computer science areas by CSRankings.[1] Fig. 2 lists the research areas, indicating the interdisciplinary nature and widespread popularity of visualizations. The primary area is Visualization, accounting for around one-third (34/98). The next areas are Human-computer Interaction and Databases. Besides, we see research efforts from Artificial Intelligence, Data Mining, Computer Graphics, and the Web & Information Retrieval.

Another finding is the increasing trend of applying AI methods for data visualization. Fig. 2 shows that the total number of publications has been increasing steadily over the last decade, particularly with a surge since 2018 and a peak at 2020. Given the wide and ever-growing research efforts and interests, we believe that this topic will receive more attention in the near future.

We acknowledge the limitations of our search methodology which is based on manual search over citation and reference graphs. Consequently, our survey should be seen as an effort in investigating the diversity of related research, providing sufficient research instances to contextualize the current research landscape, and indicating future research opportunities. As such, we do not claim comprehensiveness nor exhaustiveness. Instead, future research could augment the corpus by automated graph traversal.
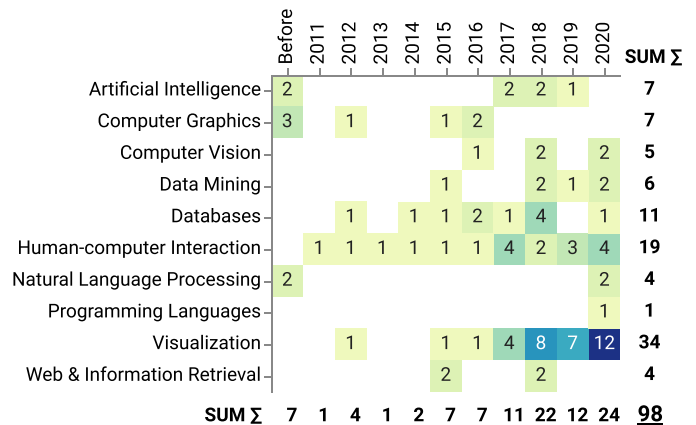
	Before	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	SUM Σ
Artificial Intelligence	2							2	2	1		7
Computer Graphics	3		1			1	2					7
Computer Vision					1			2			2	5
Data Mining					1			2	1		2	6
Databases			1		1	1	2	1	4		1	11
Human-computer Interaction		1	1	1	1	1	1	4	2	3	4	19
Natural Language Processing	2										2	4
Programming Languages											1	1
Visualization			1			1	1	4	8	7	12	34
Web & Information Retrieval						2		2				4
SUM Σ	7	1	4	1	2	7	7	11	22	12	24	**98**

Fig. 2. The number of papers per research area and per year.

3.3 Coding and Classification

Fig. 1 provides an overview of our classifications of surveyed papers, which is organized according to the what-why-how axes. This organization is well-established in visualization-related surveys [15], [26]. Nevertheless, we introduce two modifications to the "what" and "how" axes in order to contextualize our discussion in AI4VIS research.

First, from the "what" perspective, we discuss not only what visualization data is but also its representation (Section 4). For representation, we review internal representations (how visualization data is stored and operated in systems) and feature representation (how visualizations are converted into features that are mathematically and computationally convenient for analysis).

Second, from the "how" perspective, we organize approaches with a novel task abstraction. We identify seven common tasks for AI4VIS research and discuss approaches for each task in Section 6. Our motivations for such task abstractions are two-fold:

- *Reconciling techniques with system papers.* We find our corpus a mixture of technique and system papers that confound the discussion. A technique paper typically solves a single task, while a system paper could consist of multiple components, each for different tasks [17]. As such, we aim to decompose system papers into abstract tasks that allow for a collectively exhaustive taxonomy of tasks.
- *Unifying Inconsistent Vocabularies.* Due to the interdisciplinary nature of our corpus, we find tasks are usually described in inconsistent vocabularies. For instance, the task of extracting encoding choices from visualization images or specifications is described as deconstruction [1], [2] or chart mining [23]. Thus, we wish to establish a common vocabulary that enables consistent discussions for researchers from different areas to communicate the relevance and subtleties.

For the above purposes, we adopted a bottom-up approach by iteratively categorizing and labeling the tasks in surveyed papers. Thanks to the interdisciplinary nature of our corpus, we were able to start from several "seeding" tasks by referring to task taxonomy in other fields [28]. Subsequently, two authors labelled and verified whether the task could map to existing categories, and if not applicable,

1. http://csrankings.org/

discussed alternative task categorizations with the remaining authors. Conflicts of labels were also discussed until reaching a consensus. During the labeling process, we found that most tasks conformed to the "seeding" terms with few exceptions that reflected the peculiarity of visualizations. For exceptions, we discussed candidate terms in surveyed literature to select tasks that were understandable and widely communicated (e.g., recommendation). We discuss details of the task taxonomy in Section 6.

Our supplemental website, available at ai4vis.github.io, provides the details of our labels. Particularly, we label tasks at the (sub-)section level, and provide quotes from the paper to help readers understand why it falls into the task category. The website also provides an overview of the surveyed paper. In the following text, citations refer to all instances for the reported category, except where noted otherwise by phrases such as "for instance" or "like".

4 DATA: WHAT IS VISUALIZATION DATA

In this section, we formalize the concept of visualization data. Specifically, we discuss and categorize visualization data in terms of its raw data format (Section 4.1) and representations (Section 4.2). As shown in Fig. 3, we classify visualization data formats into *graphics*, *programs*, and *hybrid* that blends the benefits of both. In addition to raw data, we note visualizations are sometimes represented as carefully designed *internal representation* formats in surveyed systems. Internal representations are usually proposed to facilitate the computing by removing unnecessary information, e.g., the VQL format [29] only stores data transformation and encoding without style information. As such, internal representations are usually not exposed (outputted and shared). Finally, we review feature presentations, including *feature engineering* and *feature learning*. Feature presentations are vital for machine learning tasks by concerting visualizations into features that are mathematically and computationally convenient to analyze. We discuss them due to the increasing interest in applying machine learning to visualizations.

4.1 Data

Visualization data can be stored in different content formats such as graphics and programs. The choice of content formats directly influences the downstream operation possibly on visualization data, since different formats have own advantages and disadvantages. Here, we discuss three formats we identified in our corpus: graphics, programs, and

hybrids. As shown in Fig. 3, both graphics and programs are common in the corpus, and three recent research papers has proposed hybrid formats for the combined power.

4.1.1 Graphics

Graphics are a natural and expressive content format of visualizations, since visualizations are defined as a graphical representation of data. As such, they appear most frequently in our corpus. It is common to author and store visualization as *raster graphics* (bitmaps) for easy usage and sharing [30]. Nevertheless, raster graphics are a standalone and lossy representation which loses the visualization semantics (e.g., chart type, visual encoding, underlying data). To perform automated analysis, reverse engineering is often a pre-requisite, i.e., to reconstruct the lost information using computer vision and machine learning approaches [31], [32]. However, reverse engineering still remains an open problem with challenges to overcome in terms of robustness and accuracy [7]. In conclusion, the lossy nature of raster graphics hinders the machines from easily interpreting and transforming the visualization [7].

Vector graphics represents a less lossy alternative. They have advantages over raster graphics in that they can be scalable up without aliasing. Visualizations are usually stored in the Scalable Vector Graphics (SVG) format [30], which allows describing visual elements as shapes (e.g., rectangles and text) with styles (e.g., positions and fill-color). Those low-level descriptions reduce the difficulties of reverse engineering, e.g., it is no longer necessary to apply computer vision techniques to detect objects such as texts [33]. Besides, this format enables support for interactivity and animation. Nevertheless, high-level visualization semantics such as visual encoding and underlying data is still lost, whose extraction requires considerable effort [2], [34].

4.1.2 Program

Researchers have developed approaches for describing and storing visualizations as computer *programs*. Programs retain necessary information to construct the visualization, e.g., the underlying data. The information is usually represented by languages, which are classified into imperative and declarative programming.

Imperative visualization languages require users to specify step-by-step commands to create the visualization. For example, D3 [35] is a JavaScript-based toolkit that assists programmers in specifying graphical marks (e.g., bar, line) with visual attributes. Programs by imperative languages are open-ended and thus allow the flexible creation of visualizations. However, the open-ended characteristic renders it unstructured, with irregularities and ambiguities that hinder efficient machine-analysis from extracting the semantics (e.g., visual encodings). As such, we observe few efforts in analyzing imperative programs. In a slightly different vein, Bolte and Bruckner [36] recently proposed Vis-a-Vis, a visual analytic approach for exploring visualization source codes. The authors argued for more efforts on analyzing visualization programs.

Declarative languages ask programmers to directly describe the desired results, which is referred as specifications.
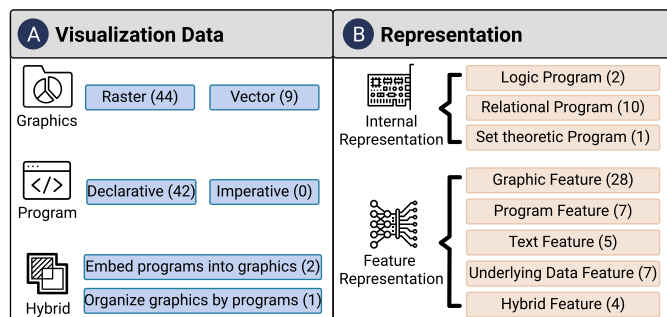

Fig. 3. Summary of categorization of (A) visualization data and (B) its representation. Numbers in the parentheses indicate the count.

Specifications (e.g., Vega-Lite [37]) encapsulate step-by-step commands for visualization reconstruction into semantic components such as data encodings, axes, and legend properties. This encapsulation is achieved by providing sensible defaults and introducing constraints with prescribed properties and structures. As such, declarative programs tend to be less or equally expressive as imperative programs, depending on their design. Since specifications contain tags or markers to separate semantic elements and enforce hierarchies, they are deemed semi-structured and thus more helpful for computer processing. It is, therefore, common to generate or collect specifications to conduct data-driven research (e.g., VizML [38]).

4.1.3 Hybrid

Recent research proposes several *hybrid* content formats that incorporate the benefits of both graphics and programs. Although such efforts remain limited, we provide our embryonic classification here, hoping to motivate future theories and models.

Two approaches aim to *embed programs into graphics*. Vis-Code [6] presents an embedding approach based on deep image steganography, that is, to conceal visualization specifications and meta information within the bitmap image. Similarly, Chartem [7] encodes information (e.g., the visualization specifications) in the background of a chart image. Both embedding techniques reduce the overhead to decode underlying visualization specifications, by showing that the encoded information can be extracted in an efficient and less error-prone manner while reducing the interference to human perception.

Loom [39] takes a different approach and seeks to *organize graphics by programs*. Loom proposes to share interactive visualizations by filling the gap between two extremes, i.e., sharing non-interactive formats such as images, and sharing the data, source codes, and software. Specifically, it formulates interface visualizations as a standalone object built on an action tree. Each intermediate node of the tree represents an interaction such as clicking, and the leaf node stores the resulting visualization image. Therefore, users can share and interact with the graphics as if they have access to the original source codes and software.

4.1.4 Discussion and Open Questions

Despite the advantages of vector graphics, they appear less often (9) than raster graphics (44), which might be due to the popularity of raster graphics in daily life. However, the lossy nature restricts the availability of machine interpretation and requires continued research on reverse engineering.

Arguably, programs are not friendly to most people except for programmers. As such, programs tend to be less shared by laypeople online, which hinders their collection and reuse. For instance, existing corpora mainly include graphics (e.g., [40], [41], [42]) or tabular datasets (e.g., [43]) instead of programs, although programs are also found to be common in our corpus (42/98). This suggests the need for more recognition of balancing the machine- and human-friendliness of visualization formats.

Visualization data often appears as byproducts in different mediums such as PDF documents [23], computational sheets, and notebooks [44]. In order for AI to access them, it needs to be able to recognize and extract visualization data, which brings challenges for interpreting the medium's structure, developing conventional standards, and proposing object detection algorithms. To that end, recent research in parsing and analyzing visualization notebooks seems a promising direction [3].

4.2 Representation

Now that we have considered different content formats of raw visualization data, the next challenge is its representation. First, raw visualization data in the format of images or programs might not explicitly represent the semantic information needed (e.g., chart type). Thus, it is usually helpful to store and operate visualization data in the internal representation formats to facilitate the process. Second, raw data need to be converted into feature representation to enable machine-learning techniques. Thus, we discuss feature engineering and feature learning approaches used to extract visualization features.

4.2.1 Internal Representation

Visualization programs tend to contain extraneous details (e.g., visual style) or miss semantics (e.g., chart type) that might not meet particular needs of research. Thus, systems usually express and operate on visualizations in simpler or more structured formats by removing unwanted or unnecessary specifications and adding customized information, which we refer to as *internal representations*. However, we only find 13 papers that explicitly propose formal internal representation formats that are designed towards the high-level goal of facilitating computation (Fig. 4).

Draco [49] (Fig. 4(A)) uses Answer Set Programming to express visualization specifications as logical facts. This falls into *logic programming* [76], which expresses problem domains as facts or rules and benefits logical computation. For instance, logical facts are used to check whether the specifications satisfy compound rules regarding the design knowledge.

Literature from data mining and databases [29], [52], [53], [55], [77], [78], [79], [80], [81], [82] usually uses *relational programming* to form visualizations as queries into database (Fig. 4(B)). Those relational queries facilitate operations on collections of visualizations such as composing, filtering, and sorting. This paradigm is also adopted by the CompassQL language in Voyager [83], [84], [85].

Finally, Wang *et al.* [75] proposed a *set-theoretic programming* representation that describes charts as a set of visual elements (Fig. 4(C)), which facilitates set computation, e.g., to determine whether a visualization is a super-set of another.

4.2.2 Feature Representation

In this section, we discuss the features of visualizations, which are the measurable properties serving as the input to machine learning models. Features are extracted by *feature engineering* or *feature learning* [86]. Feature engineering is the process of using domain knowledge to extract features from raw data, while feature learning replaces this manual process by developing automated approaches that automatically discover useful representations. The features of
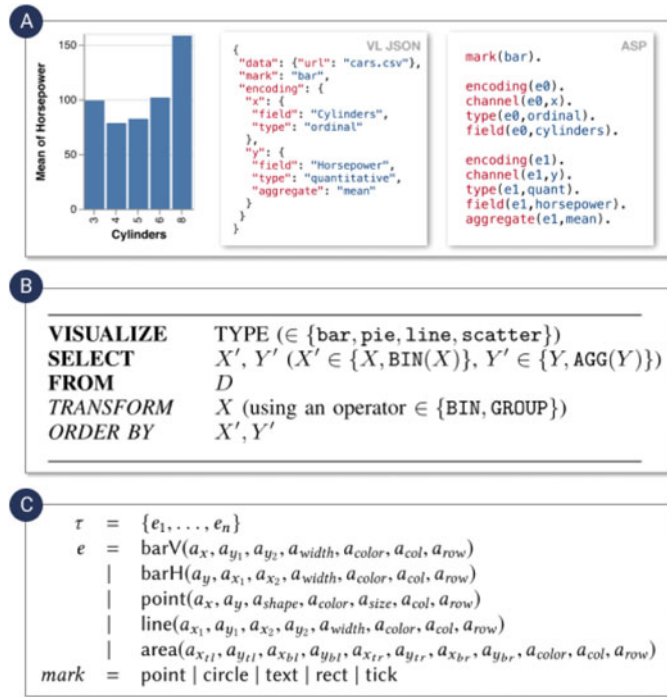
Fig. 4. Internal representations of visualizations: (A) logic programming maps specifications into logic facts for computation [49]; (B) relational programming expresses visualizations in query language for relational operations such as selection [53]; (C) set-theoretic programming facilitates set operations [75].

visualization data are multimodal, including *graphics*, *program*, *text*, and *underlying data* (Table 1). Some papers use multiple features for different tasks [31], [46] or use *hybrids* by feature fusion for improving performances [32], [45], [51], [56]. In the following text, we describe each category in detail.

Graphical features are the most common features of visualizations. The overarching goal is for predictive tasks, e.g., to predict the chart type, or detection tasks. Early work uses general image descriptors such as bag-of-keypoints [45] or patch descriptors [32]. These descriptors are designed for general visual content, containing only low-level information such as shapes and regions. To raise the level of abstraction, researchers have proposed special domain descriptors for visualization-specific information, and in many cases outperform general descriptors (e.g., [31]). Examples include regions of text elements (e.g., titles and labels) [32], positions of text and mark elements [31], [40], [46], relative positions between text and marks [47], and visual styles of elements [40]. Despite promising results, such feature engineering process remains labor-intensive and requires expertise. Besides, it remains

unclear whether human-crafted features are informative and discriminating for accomplishing machine learning tasks. Therefore, recent work has adopted feature learning more than feature engineering (e.g., 12 versus 7 in 2020). Particularly, convolutional neural networks (CNNs) have been widely used to automatically learn spatial hierarchies of image features and shown to outperform early approaches [12], [13], [31], [46], [51], [57], [58], [59], [60], [62], [63], [64], [65], [66], [67], [68], [69], [70], [71], [72]. VisCode [6] recently uses an autoencoder to learn an effective representation that could conceal additional information. Fu *et al.* [61] used the latent vectors of the autoencoder model for prediction tasks. However, these models currently face the same challenge as early general image descriptors in that they might not capture visualization-specific information. This limits their capability in high-level tasks such as assessment [61] and visual question answering [12], where the performances are relatively dissatisfactory.

Program features are extracted from the programs such as specifications. Probably the most straightforward representation is the parameters. For instance, MobileVisFixer [2] and Retrieve-then-Adapt [1] train models that learn to operate on the chart parameters, e.g., positioning the element. Burns *et al.* [48] extracted communicative signals (such as whether a group of bars is colored differently from the other bars) that they fed into a Bayesian model. Draco [49] contains constraints over facts that encode visualization design knowledge. These constraints describe whether a visualization conforms to best practices of effective visual design. However, little work uses programs as the training input which might be due to several reasons, including the lack of training data or the overheads of reserve engineering to extract the program from visualization graphics. Still, programs are a promising representation as they contain high-level visualization-specific information. For example, ChartSeer [9] converts Vega-Lite specifications of charts into visualization embeddings by autoencoders. The embeddings are used to measure similarities between charts to assist in analyzing chart ensembles, and proven effective in user studies. Their results suggest that program features are promising in semantically characterizing charts.

Text features refer to the text content in visualizations such as titles. They are considered to improve the feature informativeness by incorporating semantic information. For instance, two systems describe text information with statistical models, i.e., bag-of-words [45], [51], and incorporate the resulting text features with graphical features to improve the performance for chart detection and classification tasks. Moreover, text features can capture the subject matter of visualizations. VizCommender [3] is a recent content-based recommendation system built on machine learning models for

TABLE 1
A Summary of Feature Representation of Visualizations for Machine Learning Tasks

	Graphics	Program	Text	Underlying Data
Feature Engineering	general image descriptors [11], [32], [42], [45] element positions or regions [31], [32], [40], [46], [47] element styles [40]	parameters [1], [2] communicative signals [48] design rules [49], [50]	statistical models [3], [45], [51], [52]	statistics [38], [43], [53]–[55] one-hot vector [56]
Feature Learning	convolutional neural network [12], [13], [31], [46], [51], [57]–[61], [61]–[72] autoencoder [6], [61]	autoencoder [9], [73]	embedding models [3], [56]	autoencoder [74]

predicting semantic similarity between two visualization repositories, which shows a high agreement with a human majority vote. Chart Constellations [56] uses word embeddings to measure the similarity between charts. However, it still remains unclear how text features could be effectively fused with graphical features to more comprehensively represent a visualization, e.g., to balance text-based and style-based similarities.

The last category of visualization features is in the *underlying data* that the visualization encodes. Chart Constellations [56] describes the encoded data columns by a one-hot vector encoding, which is then used to compute chart-wise similarities. Besides such *descriptive* purposes (i.e., to describe a visualization), data features are found to be mainly *predictive* (i.e., to predict the visual encoding). Data2Vis [74] adopts a sequence-to-sequence autoencoder structure that models the input dataset in the JSON format. However, the sequence-to-sequence structure might not well capture the characteristics of the underlying data such as the data type. DeepEye [52], [53], [55] and VizDeck [54] perform feature engineering to consider data statistics such as the number of unique values in a column. VizML [38] further extends this approach to 841 features including single- and pairwise-column features of the input dataset and significantly outperforms Data2Vis and DeepEye. The analysis of VizML suggests that those features are not independent and some appear to be of little importance. Future research should propose effective feature selection approaches or complementary feature learning methods.

4.2.3 Discussion and Open Questions

Learning Visualization-Specific Features. Existing off-the-shelf computer vision or machine learning models are originally designed for general visual content or relational data. They might fall short when applied to visualizations, e.g., visualization assessment [61] and Data2Vis [74] since they do not capture visualization-specific features such as visual encodings and encoded data that are critical. Researchers have proposed feature engineering approaches to craft features, which, however, are labor-intensive without guarantees for success. With the rapid development of deep learning techniques, we envision models tailored to visualizations that effectively learn visualization-specific features.

Fusing Multimodal Features. Visualizations are unlikely to be comprehensively represented by only one feature. However, as shown in Fig. 3, most work only leverages graphic features while other modalities remain under-explored. In contrast, fusing multi-modal features could improve performances of chart detection and classification [45], [51]. Promising avenues for future work lie in leveraging feature fusion for high-level tasks such as question answering and similarity-based recommendation, which are not well-solved by the single-modal feature.

5 GOAL: WHY APPLY AI TO VISUALIZATION DATA

The goals of applying AI to visualization data cover a wide spectrum, pursued by research efforts from different areas. We adopt a deductive classification method to create a mutually exclusive and collectively exhaustive taxonomy that better structures our discussion. Specifically, we

Fig. 5. Matrix of goals of applying AI to visualization data with sub-categories. We subdivide goals along two axes deductively. Numbers in the parentheses indicate the count. A paper might have multiple goals.

subdivide goals along two axes deductively: whether visualizations are the input or output and whether visualizations are single or many. We further merge outputting single visualization and outputting many visualizations from an inductive perspective, i.e., we observe that they share the same sub-categorization. Therefore, we finally classify goals into 3 categories, which are further subdivided (Fig. 5):

- *Visualization Generation* outputs single or many visualizations given different user inputs.
- *Visualization Enhancement* processes and applies enhancement to an input visualization.
- *Visualization Analysis* concerns organizing and exploiting a visualization collection.

It should be noted that some work is a technique paper that introduces a novel algorithm without practically demonstrating the applications (e.g., [66]). We do not label the goals for those techniques but instead discuss their tasks in Section 6.

5.1 Visualization Generation

One of the central research problems in the visualization community is to ease the creation of visualizations, which is mostly studied in surveyed papers (37 papers). This is important since authoring effective visualizations is often time-consuming and challenging even for professionals [22]. As such, the ultimate goal of work in this category is to automatically generate visualizations. We identified four subcategories that distinguish visualization generation approaches by user input.

Data-based generation outputs visualizations given a database or a data-table. These approaches assist in visual data analysis and have been extensively studied over the last decades. Early research dates back to 1986 [87], while it still remains an important question nowadays. Recent work like Draco [49], DeepEye [53], VizML [38], and DataShot [88] make efforts on the direction.

Visual analysis is an iterative process where the next step of analysis often depends on earlier insights, motivating the research on *anchor-based generation*. The problem is to recommend a visualization given an anchor visualization. For instance, SeeDB [81] intelligently recommends visualizations with large deviation to the anchor visualizations, since they deem most "interesting" to users. Similarly, DiVE [79] aims to vary the visualization recommendation, while Dziban [50] targets at maintaining consistency.

Related to anchor-based generation, *design-based generation* studies the problem of generating visualizations by injecting the target data into a reference design. This is referred to as style transfer [89] or visualization-by-example [75]. Another recent example is Retrieve-Then-Adapt [1] that applies pre-defined design templates to user information.

The last category is *context-based generation*, where the input only provides some contextual information such as a natural language description [90] or news articles [58]. An important task for context-based generation is to recommend data that is mostly related to the given context.

5.2 Visualization Enhancement

The proliferation of visualizations gives rise to research efforts in enhancing the use of existing visualizations (30 papers). An important question is to *retarget* visualizations to different environments. For example, VisCode [6] and Chartem [7] encode additional information in visualization images. It is also common to summarize visualizations to generate natural language descriptions such as *captions* and *annotations*. This approach transforms visualizations from visual to non-visual modality, whereby enabling multi-modal interactions [91] or enabling people with vision impairments to consume charts [46]. Related to natural language, recent research challenges machines to perform *question answering* on visualizations, that is, to generate answers given a question (e.g., [12]). Finally, some work explores adding *interactions* to visualizations to improve the legibility and interactivity, e.g., Graphical Overlays [92] uses layered elements to aid chart reading.

5.3 Visualization Analysis

With the increasing availability of visualization data, recent research has constructed visualization databases and investigated methods for managing and analyzing these collections (14 papers). *Retrieval* has been largely studied in the field of information systems and databases, helping users search for visualizations that match their needs. For instance, Retrieve-Then-Adapt [1] assists users in finding an example chart suitable for encoding their data. Saleh *et al.* [93] developed a search engine that returns stylistically similar visualizations given a query visualization.

Another promising set of work has started to *mine* visualization collections to derive useful information such as the visualization usage on the web [40] or in the scientific literature [11], [42], as well as design patterns in visualizations [94], [95] and multiple-view systems [8]. The mined patterns provide evidence for recommending visualizations.

Related to mining, some work facilitates *visual data analysis*. Specifically, they [9], [62] consider charts to be the analytical object and propose a visual analytic approach for analyzing data from charts ensembles by providing analytical guidance for effective visual analysis, e.g., which charts to examine next.

6 TASKS: HOW TO APPLY AI TO VISUALIZATION DATA

In this section, we focus on common tasks that researchers apply to visualization data. We organize the observed tasks into seven primary tasks as follows where the number indicates the count:

- *Transformation* coverts visualization data from one modality (e.g., graphics) to another (e.g., program) (35/98).
- *Assessment* measures the absolute or relative quality of a visualization in terms of scores or rankings (19/98).
- *Comparison* estimates the similarity or other metrics between two visualizations (12/98).
- *Querying* finds the target visualization relevant with a user query within visualization collections (10/98).
- *Reasoning* challenges AI to interpret visualizations to derive high-level information like insights and summaries (22/98).
- *Recommendation* automates the creation of visualizations by suggesting data and/or visual encodings (38/98).
- *Mining* discovers insights from visualization databases (7/98).

Most of those tasks originate from well-known terminology. Transformation, assessment, and visual reasoning are well-studied tasks in the field of computer vision [28], while querying and mining come from database and information system research. Two exceptions are comparison and recommendation. Although the comparison is similar to the image similarity search task in computer vision [28], we find a large body of visualization research studying other metrics (e.g., difference) between two visualizations and thus decide the wording comparison. Recommendation is a widely studied task in the visualization literature (e.g., [21], [83]).

We obverse relationships between tasks and published venues. For instance, the Visualization community has mainly studied recommendation (17/34), while Computer Vision and Artificial Intelligence research focuses more on transformation (8/16) and reasoning (7/16). This shows the diversity and wide attention among different research communities.

6.1 Transformation

Transformation is the operation that converts the content formats of visualizations. Particularly, it is straightforward to transform visualization programs into graphics by visualization tools or libraries (e.g., [35], [37]). A more challenging problem is the reverse process, i.e., reconstructing programs from graphics. This process is also known as reverse engineering [31]. In the following text, we focus on the reverse engineering problem.

Relations to Goals and Other Tasks. Transformation is usually the first task for visualization enhancement and analysis, especially when the input is images. As such, it is often a prerequisite for remaining tasks. For instance, the extracted information can be used for querying [94] and reasoning [96].

Relations to Visualization Data. Several works study the problem of transforming visualization images into another image by altering the data or visual styles [1], [89], [94]. Nevertheless, their approaches are built on reverse engineering, i.e., to extract the encodings first and then replace the data. Little research explored the direct transformation in the image space [6], [7], [97], [98]. As such, we do not dedicate a separate discussion on image-to-image transformation at the current stage. Future research could extend our taxonomy with further development in this field.

6.1.1 Challenges and Methods

Visualization reverse engineering has been widely and extensively studied over the past decades. Early research could date back to 2001 when Zhou and Tan [99] proposed a learning-based paradigm for chart recognition. Since then, much research has been devoted to extracting semantic information from visualization images such as chart types, visual encoding, and underlying data. Ideally, reverse engineering is expected to yield cycle-consistency; that is, its output should be able to re-generate the original visualization. Despite promising preliminary results, several challenges remain to be overcome since much work makes simplifying assumptions on the expected input or output. For instance, several approaches take vector graphics as input [2], [11], [34], [40], [89], [94], [100], [101], [102], assuming the type of each visual element is available as SVG. Most work is limited to a predefined set of chart types, while little work [2], [34], [58], [59], [89], [94], [100], [103] applies to more bespoke visualizations. Besides, a large body of research focuses on extracting a particular portion of information, such as chart classification [42], [51], [58], [59], [63], [64], [65], object separation [104], [105], and object clustering [2], [100], [101].

We summarize a conceptual framework of the reverse engineering process to provide an overview of existing approaches and identify research gaps. We developed the framework via a bottom-up approach, where we abstract existing methods, identify their simplifying assumptions, and iteratively merge the results. As shown in Fig. 6, we identify two distinct phases. The first phase decomposes visualization graphics into semantic elements (e.g., axis, mark) through machine learning and computer vision techniques, including object detection, classification, and clustering. The second phase performs mathematical computation over the decomposed semantic elements to extract visual encoding and/or the underlying data. In the following text, we describe each phase.

Decomposing. The decomposing phase varies depending on the input (Fig. 6A). The primary step for raster graphics is to detect and classify visual elements such as text and shapes. This is approached by traditional image processing techniques (e.g., edge detection, morphological operations) in early work [32], [46], [47], [109] and machine learning or deep learning approaches (e.g., Mark RCNN) in work published in 2015 and later [10], [31], [45], [46], [60], [62], [91], [103], [110], [111]. This element recognition step faces chart-specific challenges, e.g., to cope with visual clutter in line charts and scatterplots [104], [105]. In addition, since existing object detection models are prone to rotation that is common for pie sectors, Choi *et al.* [46] proposed a special heuristic to pie charts by grouping the nearby pixels with the same color. The output of this element recognition step is usually the position and class of each visual element, which are already available in the SVG specifications of vector graphics. In other words, vector graphics remove the overhead of element recognition.

Chart detection and classification are an another step of the decomposing phase. This step faces two important choices: the classifier and the feature representation. Classical classifiers (e.g., support vector machine) [32], [40], [45],
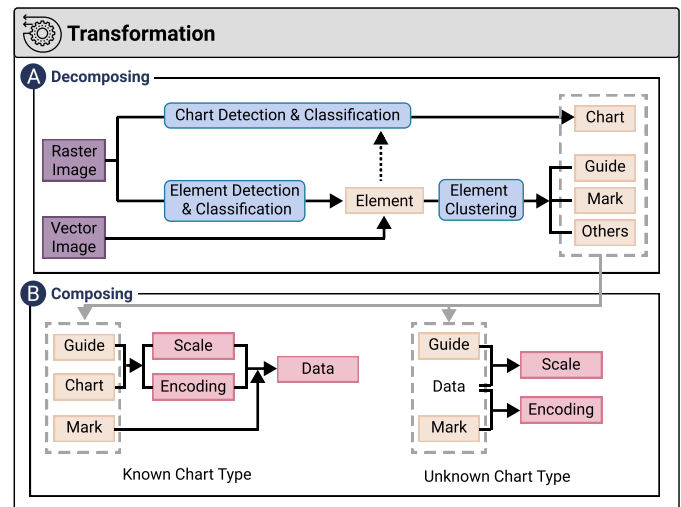

Fig. 6. The conceptual framework of the reverse engineering process: (A) It first decomposes input graphics into semantic groups such as guides (axes and legends) and marks; (B) The resulting information is fed into mathematical computation to extract the visual encoding and/or underlying data, depending on whether the chart type is known. It remains an open challenge to derive both visual encoding and data from bespoke, unknown chart types.

[112] have been gradually superseded by deep learning classifiers (e.g., convolutional neural network (CNN)) [42], [46], [51], [58], [59], [60], [62], [63], [64], [65] in visualization classification tasks. This is mainly because CNNs can effectively learn abstract features from raw visualization images, while classical classifiers require hand-crafted image features such as histograms of the image gradients (HOG) [112] and dense sampling [32], [45]. Several approaches seek to improve the representativeness of features by incorporating element-level features such as text [32], [51] and shape style features [40]. The last step of the decomposing phase addresses element clustering, that is, to cluster visual elements into semantic groups, including guides (axes and legends), marks, and other information such as annotations. This step is typically separately discussed for text and shape elements. On one hand, clustering text is usually formalized as a classification problem, that is, to classify and group text according to their text roles such as x-axis-label and legend-title [10], [31], [32], [45], [46], [47], [62], [91], [111], [113]. On the other hand, this classification-based approach is not always readily applicable to shape clustering, since the roles of shape depend on the chart type. As such, researchers usually simplify this problem by focusing on common charts where shapes are well-defined. For instance, Poco and Heer [31] trained a classifier to detect area, bar, line, and plotting shapes, and consequently grouped shapes by types. To support customized visualizations, several approaches [2], [34], [89], [94], [100] use the node hierarchy information to group shape nodes under the same ancestor. Nevertheless, those approaches are only applicable to vector graphics. Finally, the shape clusters are associated with the text clusters to identify axes, legends, and label-mark relationships.

Composing. After the visualization graphic has been decomposed into semantic groups (e.g., guides and marks), the *composing phase* (Fig. 6B) aims to extract the visual

encoding and/or the underlying data from this semantic information. Different from the decomposing phase that uses computer vision and machine learning tasks, this composing phase mainly uses heuristics by leveraging domain knowledge about visualizations. We find two common methodological themes of those heuristics, depending on whether the chart type has been extracted from the previous phase.

The first class of heuristics uses information about the chart type and guides to determine the scale and the encoding, which is dominant in our corpus [32], [45], [46], [47], [60], [62], [91], [109], [110], [111], [113]. For instance, given a scatterplot with axes and legends, it is straightforward to derive the visual encodings, i.e., the x/y positions maps to numerical values and the color encodes the categorical data, and to calculate the scale. Consequently, the underlying data could be computed via applying the reserve scale computations over the marks. However, those heuristics are often limited to a small set of chart types.

Another class [34], [89], [94] studies the more challenging problem of decoding bespoke visualizations where the chart type is unknown. However, they focus on D3 charts where the underlying data is available by crawling the SVG node on the web. In this way, they develop heuristics to determine the scale from guides and data, and to derive the encoding from data and marks.

6.1.2 Discussion and Open Questions

In conclusion, it remains an open challenge to derive both the visual encoding and the underlying data from bespoke visualization graphics, whose chart types are not limited to common ones. An important future research direction would be to improve the current heuristics for determining visual encodings from bespoke visualizations, i.e., by machine-learning approaches.

Another primary challenge of reverse engineering lies in robustness and accuracy. The pipeline of reverse engineering usually consists of multiple sequentially dependent tasks that are prone to single points of failure; that is, the failure of one task would spread to the whole system. For instance, common failure cases such as text detection would impede the extraction of guides and consequently the visual encoding [31], [91].

This motivates the use of semi-automatic approaches that address imperfect algorithms with human intervention [92], [114], [115]. Nevertheless, those frameworks of incorporating automatic algorithms with human intervention are specific to chart types and therefore not readily applicable to more general situations. A related research direction is to investigate a general framework for deconstructing bespoke visualizations.

6.2 Assessment

There is a long history of research on teaching machines to assess and rank the quality of data visualizations. As shown in Table 2 and Fig. 7, assessment outputs a numerical score of the visualization quality or measures the relative quality in terms of ranking.

Relations to Goals and Other Tasks. The key motivation of assessment is to improve visualization design, e.g., to

TABLE 2
The Assessment Task is Classified by the Output (row) and the Method (column)

	Rule-based	Machine Learning	Hybrid
Rankings	effectiveness rankings [87]	learning-to-rank [52], [53], [55], [58]	
Scores	convert rankings to scores [49], [84], [85] hand-crafted metrics [2], [5], [6], [77], [78], [90], [106], [107], [108]	learning-to-rank with scores [1], predictive regression [43], [54], [61]	learning-to-weight hand-crafted metrics [49]

derive scoring metrics that can be used as cost functions for automatic generation. That said, assessment is often combined with recommendation.

Relations to Visualization Data. Most surveyed techniques take visualization programs as input, focusing on the visual encoding and data quality. Nevertheless, Fu *et al.* [61] propose an approach for assessing visualization images.

6.2.1 Challenges and Methods

Assessment is challenging due to the human-centered nature of visualizations that requires large-scale empirical experiments to understand what makes a visualization "good". However, the knowledge derived from large-scale empirical experiments is often represented as design guidelines instead of quantifiable rules. As such, much research aims to quantify knowledge about "good" visualizations. In 1986, Mackinlay [87] developed the APT system that *ranked* the effectiveness according to the accuracy rankings of quantitative perceptual tasks for different visual encoding channels.

However, this ranking-based approach only reflects the relative quality of visualizations. *Scoring-based* approaches are often more desirable since scores measure the absolute quality and therefore benefit down-streaming tasks, e.g., scores can be used as the cost function for optimization. To that end, Voyager [84], [85] and Draco [49] map different single-criteria rankings to numerical scores. Besides, researchers often leverage domain knowledge to design hand-crafted, *rule-based* metrics that measures the visualization quality such as informativeness [90], interestingness [77], [78], [106], accuracy [77], [78], significance [107], saliency [108], visual importance [6], complexity [5], and mobile-friendliness [2]. However, designing hand-crafted metrics usually requires considerable effort. More critically, the design process is often unsystematic and lacks a strong methodological base. For example, Wu *et al.* [2]
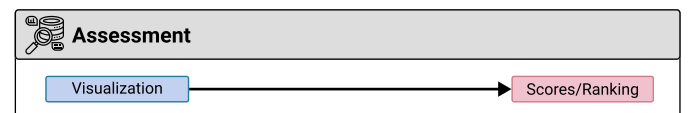

Fig. 7. Input-output model of assessment.

demonstrated that even seemingly reasonable metrics do not always survive experimental scrutiny. Besides, questions arise about how to weigh scores to reflect the overall, multi-criteria quality. Several systems [5], [77], [78], [84], [85], [90] determine the weights for each score through manual refinements that could become unsystematic.

As such, another line of research seeks to propose more systematical *machine learning* approaches that learn to rank and/or score visualizations from data collected from empirical studies. VizByWiki [58] and DeepEye [52], [53], [55] formulate a learning-to-rank problem that learns to rank visualizations from crowdsourced data. Retrieve-then-Adapt [1] extends the learning-to-rank model that simultaneously outputs paired scores. VizDeck [54] learns a linear scoring function from users' up- and downvotes. VizNet [43] demonstrates the feasibility of training a machine-learning model to predict the effectiveness of visual encodings.

Nevertheless, those machine-learning approaches face two major challenges, including poor generalisability and explainability. First, the aforementioned models are trained over statistical or visual encoding properties, assuming the underlying dataset and specification is available. More importantly, they only support a limited number of chart types. To that end, Fu *et al.* [61] propose a more general approach to assess the quality of visualization images that generalizes to different visualization type and does not require additional information except the visualization images. Second, machine-learning models lack explainability that might decrease trust. Moreover, they often exclude knowledge derived from empirical studies. To address these limitations, Draco [49] takes a *hybrid* perspective by encoding design knowledge as constraints and learning a weighting function to trade off those constraints, whereby outputting a final score.

6.2.2 Discussion and Open Questions

Existing approaches predominantly focus on objective qualities that can be measured via user studies (e.g., task performance and competition time). However, subjective metrics such as aesthetics are relatively underexplored, despite that they are considered as important features of good visualizations [116]. This is challenging since it is difficult to harvest crowdsourced data of subjective quality of visualizations, since crowdsourced judgments can be inconsistent and inaccurate. This underscores research needs to propose methods for generating large-scale training datasets for visualization research in a reliable and sustainable manner. One promising way is to incorporate expert knowledge and crowdsourcing experiments in dataset generation.

Going forward, we envision machine-learning approaches that not only assess the visualization but also provide insightful explanations. In this way, the approaches embrace explainability by translating ML models into human-readable explanations and even useful design guidelines.

6.3 Comparison

Characterizing the similarity or other metrics between two visualizations is helpful when dealing with a visualization collection (Fig. 8).
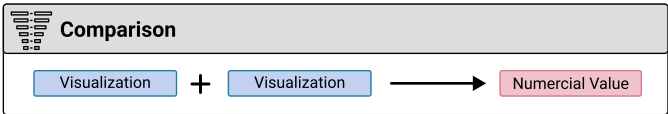

Fig. 8. Input-output model of comparison.

Relations to Goals and Other Tasks. Comparison is found to assist in visualization generation and analysis. The comparison metrics can be used as cost-functions in 1) recommendation to perform anchor-based visualization generation (e.g., [81]), as well as 2) querying to perform "query-by-example" (e.g., [8]). Besides, the assessment metrics can be used to compute the difference.

Relations to Visualization Data. Comparison is studied on both visualization programs and graphics, as shown in Table 3.

6.3.1 Challenges and Methods

Perhaps the most straightforward approach for comparing two visualizations is to calculate the *difference*. GraphScape [119] is a directed graph model where each link represents an edit operation (e.g., add field) and nodes denote the resulting visualizations. Subsequently, each edit operation is registered with a cost, which is learned from human judgments. Dziban [50] further translates the graph model into a set of constraints and weights, similar to Draco [49]. In this way, both approaches explicitly model the difference between two charts as an operation associated with a numerical cost. Nevertheless, this difference-based approach becomes sophisticated when there exist multiple, often under-specified, operations between two charts, where graph traversal is essential for searching and weighting all possible paths. This is further complicated by the limitation that GraphScape only includes operations regarding data transformation and visual encodings. Although it is methodologically feasible to extend GraphScape to support other operations such as recoloring, such extensions are labor-intensive, without a guarantee for exhaustiveness.

Partly due to the above challenges of difference-based methods, most research adopts **distance-based** measurements [3], [8], [9], [29], [52], [56], [79], [81], [93], [117], [118]. The key idea is to convert a visualization into a feature vector, and compute the distance between two feature vectors according to distance functions. Thus, the technical challenges of distance-based measurements are two-fold: the choice of features and the distance function.

The features of visualizations vary among the underlying sources, as discussed in Section 4.2.2. In the context of comparison, we identify four primary sources including

TABLE 3
Comparison is Classified by the Method (row) and the Input (column)

	Program	Text	Data	Graphics
Distance	[9], [52], [79]	[8], [93]	[3], [56]	[9], [29], [52], [79], [81], [117], [118]
Difference	[50], [56], [119]			

graphics, text, data, and specifications. For instance, Saleh *et al.* [93] extract low-level visual features from *graphics* to learn style similarity, and Chen *et al.* [8] model the configuration pattern of multiple-view visualization systems as a 1×126 vector measuring the layout. Regarding *text* features, VizCommender [3] uses both hand-crafted features (e.g., TF-IDF) and learned features (e.g., Doc2Vec). More work uses hand-crafted features for the *data* [29], [81], [117], [118]. Finally, ChartSeer [9] uses deep learning approaches to convert Vega-Lite *specifications* into embeddings.

The derived features are subsequently fed into a distance function to derive the distance. Examples of common distance functions include mutual information [8], [117], Earth Mover's Distance [29], [81], Bhattacharyya coefficient [118], and Jaccard coefficient [79]. Nevertheless, the process of selecting distance functions is hardly detailed in the literature, leaving rationales and insights unexplored. This is worsened by the potential downside of distance-based measurements that the feature representation is less interpretable than the operations in difference-based measurements. Thus, it is often difficult to interpret the results, and the user study suggests that the similarity measurement "does not fully understand their (users') intent" [9].

An important question that then arises is how to measure the overall similarity when combining multiple sources. Naive feature concatenation is a natural way to combine features from different sources [29], [79]. For instance, Luo *et al.* [52] propose a feature vector concatenated from five aspects, each represented by a one-hot vector describing visualization types, x-axis, y-axis, group/bin operations, and aggregation functions. Another method is to compute the aggregated distance by weighting *hybrid* distances, e.g., chart encoding distance, keyword tagging distance, and dimensional interaction distance by Xu *et al.* [56]. Nevertheless, both feature concatenation and distance aggregation assume a linear relationship among different vectors, which seems far from capturing the real-world complexity and thus yield limited performances when perceived by users.

6.3.2 Discussion and Open Questions

Comparison and assessment are closely related and share the same goal of outputting a numerical score. Comparison has been predominately rooted in feature engineering and hand-crafted distance functions. The major downside is that it does not actually learn from user feedback and thus usually fails to meet the users' intent. Unlike assessment, few little machine learning approaches have been applied to comparison. Several approaches (e.g., [56]) use pre-trained ML models to perform feature learning. However, they do not fine-tune the models on user feedback data. Thus, proposing dataset and ML approaches for comparison is a clear step to improve the performance.

Nevertheless, it is non-trivial to adapt ML approaches to comparison, since comparison involves two visualizations while standard ML models only take one entity as input. ScatterNet [120] addresses this issue. It is a deep learning model for predicting similarities between scatterplots by learning from crowdsourced human feedback data. Nevertheless, it is unclear how to adapt this approach to other statistical charts. A key challenge is that CNN models used in
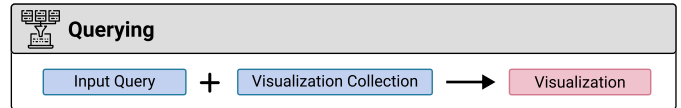

Fig. 9. Input-output model of querying.

ScatterNet are worse at capturing human perception in other charts [61], [66].

6.4 Querying

Querying is the task of retrieving relevant visualizations that satisfy the users' needs from a visualization collection (Fig. 9). It is a crucial component of Information Retrieval (IR) systems, which are also known as search engines, especially in the context of the web [121]. Querying in this context is distinct from visualization query language (Fig. 4(B)). The latter specifies visualizations as a query into a database, while the former describes a query into a visualization collection.

Relations to Goals and Other Tasks. Querying is mainly for visualization retrieval (Table 4). It is often built upon other tasks like transformation (e.g., [94]) and comparison (e.g., [8]).

Relations to Visualization Data. Querying directly in the image space can be difficult since semantic information is lost. As such, it is often performed on visualization programs, where semantic information such as titles and axis labels are available.

6.4.1 Challenges and Methods

There are two viewpoints that characterize querying: how to specify users' needs and how to match the needs.

The simplest form of querying syntax is *keywords*. Keywords are popular since they are intuitive and easy to express. Choudhury and Giles [11] developed a search engine that allowed users to search figures by keywords in the captions. Similarly, Voder [122] supports keyword-based queries into data fields as well as general words like 'outlier' from the data fact associated with a visualization. Li *et al.* [123], [124] extends keywords to *natural language* queries by extracting structural keywords from queries and matching the extracted words with text in visualizations. Nevertheless, keywords-based queries often fail to disambiguate unstructured queries since words have multiple meanings.

Structural queries are a mechanism for resolving ambiguity and improving the retrieval quality and have been used in multiple systems [1], [8], [10], [60], [94]. They are built on keywords with the addition of structural constraints. For instance, DiagramFlyer [10] is a search engine where a query contains eight key structural fields (e.g., type, x-label, legend) that can uniquely describe a visualization. Those

TABLE 4
Querying is Classified by the Method (row) and the Input (column)

	Keywords	Natural Language	Structural	Example
Exact			[8], [60], [94]	
Best	[11], [122]	[123], [124]	[1], [10]	[93]

structural constraints make it possible to search information more than text in visualizations. For instance, Retrieve-then-Adapt [1] retrieves infographics based on a query composed of graphical and textual elements. Notably, visualization specifications are a functional candidate for structural queries. Hoque and Agrawala [94]'s search engine indexes D3 visualizations as Vega-Lite specifications and supports queries in the Vega-Lite syntax. Accordingly, it lets users find visualizations based on a wide range of constraints such as mark types, encodings, and non-data-encoding attributes. In the best case where the input query is a complete Vega-Lite specification, their research engine actually supports "query-by-example".

This *example-based* query is another format that offers an intuitive method for users to specify their intent. Saleh *et al.* [93] implemented a search engine for stylistic search over infographic corpora by returning stylistically similar images given a query image. However, more sophisticated analysis methods are necessary to capture characteristics beyond stylistic similarities.

Now that we have discussed the query syntax, the next challenge is how to reason which visualizations are most relevant to the user-input query, which can be classified into *exact-match* and *best-match* methods. Exact-match techniques are used for filtering visualizations by strict conditions, e.g., to retrieve visual analytic systems containing four views [8], bar charts [94], or charts describing a dataset [60]. However, exact-matching is not always possible especially when the input conditions are too strict. As such, more systems use best-match approaches [1], [10], [11], [93], [122], [123], [124] that rank visualizations according to metrics. Those metrics measure the degree to which a visualization is relevant to the input query. They use natural language models for text components (e.g., synonyms [10], [122], text relevance [123], [124]) and similarity metrics [1], [8]. However, relevance or similarity metrics are insufficient for retrieving visualizations that are not only relevant but also of high quality. In response, Retrieve-then-Adapt [1] learns the distribution of visual elements in the corpus in an attempt to empathize visualizations with common elements, assuming that the more frequent an element is, the better it is.

6.4.2 Discussion and Open Questions

Research on indexing visualizations has been relatively limited in the past decade, leaving room to boost technical development. Due to the diversity of visualizations, even state-of-the-art methods are often restricted to certain types of visualizations, e.g., proportion-related infographics [1] or basic D3 visualizations [94]. How to generalize those approaches to more types of visualizations is a non-trivial issue that requires a deeper understanding of how visualizations should be indexed. For instance, Hoque and Agrawala's approach [94] indexes D3 visualizations in Vega-Lite syntax, which is insufficient in expressing user queries such as "sunburst diagrams". Indexing becomes more challenging when applying to raster graphics (bitmap images) that deserve research efforts, e.g., reverse engineering.

The intention gap is another challenge for querying visualizations. From a theoretical perspective, there are not enough

Fig. 10. Input-output model of reasoning.

empirical studies to understand the user needs in searching for visualizations. Such studies are formative approaches to motivate the design space of the indexes of visualizations. In a related venue, Oppermann *et al.* [3] recently found that information seeking is a core task when browsing visualization repositories, and that users were more interested in content rather than styles. Similar studies are needed to survey users and inform the design of future search engines for visualization. From a practical perspective, it is crucial to design convenient query interfaces that assist users in specifying their intent. For instance, it is easy for users to specify the region of interest on an example visualization and interactively browse the results. Hoque and Agrawala's query-by-example feature seems a promising start. We also notice a large body of research that studies query-by-pattern or query-by-sketch in time-series visualizations, e.g., [125], [126]. Future work could generalize those query paradigms to all visualization types.

6.5 Reasoning

As shown in Fig. 10, reasoning challenges machines to "read charts made for humans" [16]. Reasoning requires interpreting visualizations to derive high-level information such as insights beyond extracting visual encoding and data via reverse engineering. Reasoning is distinct from assessment since reasoning usually outputs semantic information (e.g., insights, text summaries) rather than a numerical score. As shown in Table 5, we find three common classes depending on the targeted output of the reasoning process: visual perceptual learning, chart summarization, and visual question answering. We first describe each class separately, followed by an organized discussion of existing methods and research gaps.

Relations to Goals and Other Tasks. Reasoning is mainly for visualization enhancement (e.g., summarize natural language descriptions [73]). It sometimes relies on reverse engineering to improve the algorithm performance (e.g., [96]).

Relations to Visualization Data. Reasoning is studied on both visualization images and programs.

6.5.1 Challenges and Methods

Visual perceptual learning aims to solve visual tasks by analyzing visual information. For instance, Temporal Summary

TABLE 5
Reasoning is Classified by the Class (row) and the Method (column)

	Rule-based	Learning-based
Visual Perceptual Learning	[106]	[57], [66]
Chart Summarization	[45], [48], [55], [127], [128], [129]	[13], [68], [73], [130]
Visual Question Answering	[47], [96]	[12], [67], [69], [70], [71]

Images [106] automatically extracts points of interest in charts according to predefined heuristics. Recently, deep learning methods, trained on labeled datasets, have been applied to improve machine perception of visualization images. Bylinskii *et al.* [57] presented neural network models to predict human-perceived visual importance of visualization images. Similarly, Haehn *et al.* [66] evaluated the performances of CNNs on Cleveland and McGill's 1984 perception experiments [131] and concluded that off-the-shelf CNNs were not currently a good model for human graphical perception. Their initial results underscore the importance of continued research to improve the performance.

Chart summarization becomes increasingly important with the rapid popularization of visualizations. Most existing approaches generate text summaries such as natural language description or captions [45], [48], [55], [73], [127], [128], [129], [130]. The simplest approach is to provide a short description of how to interpret the chart [55], [127], e.g., "This chart shows the trend of average departure delay in January". More advanced approaches focus on explaining and communicating high-level insights conveyed by charts. Those approaches extract the data patterns and subsequently convert patterns to natural language according to pre-defined templates [45], [48], [128], [129], e.g., "X was the most/least frequent sub-category in A". Liu *et al.* [130] offered an alternative perspective that learns most noteworthy insights with deep learning approaches. A common limitation of the above work is that natural language summaries are generated via pre-defined templates and therefore confined to few variations and generality. Thus, recent research proposes several end-to-end deep-learning solutions for generating chart captions [13], [68] or text summarization [73]. However, summarization still remains highly under-explored since the algorithm performances have much space for improvements.

Visual question answering is another emerging research area that aims to answer a natural language question given a visualization image. Traditional methods [47] first decode visualizations into data tables and then parse template-based questions into queries over the data table to generate answers. Kim *et al.* [96] recently improved a natural language parser to support free-from, crowdsourced questions. Another line of research studies end-to-end deep learning approaches [12], [67], [69], [70], [71]. The key challenge is that answering questions for visualizations requires high-level reasoning of which existing visual question answering models are not capable [12], [69]. Besides, visualization images are sensitive to small local changes, i.e., shuffling the color in legends greatly alters the charts' information. Therefore, the major problem is how to learn the features from visualization images and fuse them with features from natural language questions. DVQA [12] learns and fuses features via a sophisticated model containing multiple sub-networks, each responsible for different components such as spatial attention. Later work expands this work with improvements to the models, e.g., PlotQA [70] and LEAF-QA [67] explicitly apply reverse engineering to retrieve visual elements and feed the extracted information into sub-networks. However, there still lacks real-world datasets, good models, and evaluation metrics.
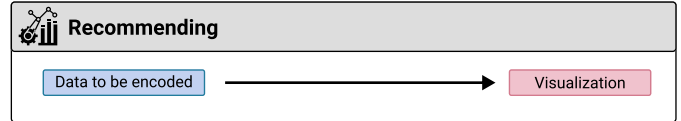

Fig. 11. Input-output model of recommendation.

6.5.2 Discussion and Open Questions

The reasoning task is currently undergoing changes since machine learning approaches, particularly deep learning models, are increasingly used. This change may attribute to the rapid advancement and successful applications of deep learning in visual reasoning. In the visualization context, research gaps emerge since off-the-shelf models for natural images are often shown to yield dissatisfactory performances on visualizations (e.g., [12], [66]). This gap is not surprising since visualizations contain relational information that is sensitive to small details that are not commonly present in natural images, i.e., a local change to a bar shape might significantly impact the encoded data and conveyed meanings. This research area remains largely under-explored. First, limited datasets are available, which hinders model development and validation. For instance, most existing datasets for visual question answering contain synthetic questions and charts, which are far from being representative for the actual use. Second, it would be pertinent to study feature learning models that are tailored to visualization images. The recent trend of decomposing end-to-end models to structures containing multiple sub-networks might be a promising method (e.g., [70]).

6.6 Recommendation

Recommendation is an important step for automating the creation of visualizations Fig. 11. As shown in Table 6, there are three methods for recommending visualizations [83]:

- Data recommendation suggests interesting data, insights, or data transformation be visualized from a database.
- Encoding recommendation determines the visual encoding (including both data and non-data encodings) given the data or other visualization elements.
- Hybrid recommendation decides both data and encodings.

Relations to Goals and Other Tasks. Recommendation is mainly for visualization generation. It is related to assessment

TABLE 6
Recommendation is Classified by the Class (row) and Methods (column)

	Optimization	Prediction
Data	[4], [29], [79], [81], [88], [117], [122], [132], [133]	
Encoding	[1], [2], [4], [5], [49], [50], [59], [84], [87], [90], [95], [106], [117], [122], [128], [134], [135], [136], [137]	[38], [74], [88]
Hybrid	[4], [52], [53], [55], [83], [85], [88], [117], [128], [135], [138]	

and comparison since the derived metrics are used as cost functions.

Relations to Visualization Data. Recommendation outputs visualization programs and subsequently visualization graphics.

6.6.1 Challenges and Methods

Data Recommendation. Given a dataset, one step of recommending visualizations is to select data fields to be visualized, and when applicable, corresponding data transformation as well. The simplest approach to decide fields is enumeration. For instance, Voyager [84], [85] enumerates all possible fields according to a predefined display order by the type and name. DataSite [128] improves this enumeration approach by computing and communicating data facts associated with the selected fields according to pre-defined templates (e.g., "Correlation of A was found between X and Y" if selected data fields are X and Y). However, enumeration imposes a heavy burden on users that motivates other research to recommend the most useful data facts, also known as insights.

Recommending insights is often approached by proposing a taxonomy of insight types (e.g., extrema), each type associated with an assessment metric. Examples are Foresight [132], Profilier [117], Voder [122], DataShot [88], VisClean [29], and Calliope [4]. One key challenge that then arises is the assessment metric. Voder [122] introduces threshold-based heuristics that classify data facts into different tiers, while the remaining systems propose various cost functions that better capture the differences between data facts. Remarkably, QuickInsights [133] propose a unified formulation of insights and scoring metrics irrespective of the type. In the context of anchor-based generation where the goal is to recommend data that meets some criteria with respect to anchor data, the aforementioned assessment metrics are augmented by or replaced by comparison metrics in VisClean [29], DiVE [79], and SeeDB [81]. For instance, SeeDB [81] recommends data by deviation with an anchor visualization.

Given the above metrics, the next challenge is to compute the best or top-k insights. This is challenging since the space of data facts grows exponentially with the number of data table columns. As such, researchers have proposed multiple strategies to speed up the computation. For instance, Foresight [132] uses sketching to quickly approximate the costs. Other systems introduce efficient searching algorithms to recommend the top-k insights. Those searching algorithms are primarily progressive or iterative [4], [29], [79], [81], [88], outputting intermediate solutions that approximate the optimal one. That said, this data recommendation problem has not yet been formulated as a prediction problem that is solved by machine-learning models, probably due to the lack of labeled training data. Notably, two approaches explicitly adopt tree-based algorithms [4], [29], leveraging the idea of GraphScape [119] that the visualization design space can be modeled as a graph for greedy or dynamic programming.

Several systems recommend data according to other input beyond a database. Particularly, those inputs are related to natural language that is beyond the core scope of this survey. Examples include natural language statements in Text-to-Viz [90], news articles in VizByWiki [58], keyword queries [55], and natural language interfaces (e.g., NL4DV [136] and FlowSense [139]).

Encoding Recommendation decides data encodings and/or non-data encodings for styling (e.g., positions).

Data encodings are extensively studied in the literature. Early approaches date back to 1986 where the APT system [87] enumerates the visual encoding space and selects the "best" encodings according to the assessment ranking in terms of expressiveness and effectiveness. This ranking-based recommendation is implemented and extended in later systems such as ShowMe [134] and Voyager [84]. In addition to ranking, several systems propose heuristic rules to decide visual encodings given the insights extracted from data recommendation or visual tasks, e.g., extreme insights or finding extremes are mapped to histograms or scatterplots. Examples include DataVizard [135], DataSite [128], Profiler [117], NL4DV [136], Calliope [4], VizAssist [137], and Voder [122].

The above heuristic-based data-encoding recommenders have recently been superseded by machine learning approaches due to the increasing availability of datasets. Draco [49] and Dziban [50] learn to assess visualizations by weighting design rules for visualizations. Subsequently, they formulate a constraint optimization problem to recommend the best or top-k visualizations. As discussed, such approaches combining assessment and optimization are commonly adopted for recommending insights. Other ML approaches seek to directly learn the mappings between data and visual encodings by training an end-to-end model, including Data2Vis [74], VizML [38], and DataShot [88].

In addition to data encodings, other approaches study how to recommend non-data-encoding attributes such as layouts and colors. Most approaches formulate optimization problems with the primary goal to define the optimization target, that is, the assessment metrics. Several metrics are human-crafted cost functions [2], [5], [59], [90], [106], while other metrics are data-driven, including machine learning models trained on human assessment dataset [1] and distances to common patterns mined from a corpus [95]. Several systems contribute novel optimization algorithms to improve efficiency, including reinforcement learning [2] and Markov chain Monte Carlo methods [1]. Similar to recommending top-k insights, both algorithms are progressive.

Hybrid Recommendation decides both data and encodings. A straightforward approach for hybrid recommendation is to combine data and encoding recommendation sequentially. This approach is widely implemented in visualization recommenders including DataVizard [135], DataSite [128], Profiler [117], Calliope [4], Voder [138], and DataShot [88]. Other approaches take an end-to-end perspective, formulating the recommendation tasks as an optimization problem. Examples are Voyager [83], [85], DeepEye and its extensions [52], [53], [55]. Thus, the core problem is to provide an overall assessment score regarding both the data and encodings. The first step is to translate a visualization into a formal representation, i.e., visualization query language (VQL). Assessment metrics are then proposed to evaluate the quality of a VQL representation. Finally, Voyager ranks the recommended visualizations, while DeepEye proposes efficient algorithms to generate the top-k visualizations.

TABLE 7
Mining is Classified by the Class (row) and the Method (column)

	Statistics	Clustering	Visual Analysis
Design Pattern	[8], [11], [40], [42], [94]	[95]	
Data Pattern			[9], [56]

6.6.2 Discussion and Open Questions

An ongoing discussion in recommending visualizations is the problem formulation including optimization and prediction. Optimization requires the careful design of optimization functions which are often the assessment scores. Another important concern is the efficient algorithms for solving the complex, sometimes multi-objective, optimization problem due to the huge design space of visualizations. Generally speaking, optimization-based approaches have the potential to be extended to the human-in-the-loop approaches, since the predicted assessment scores help users determine the visualization quality. Nevertheless, hand-crafted cost functions for assessment are often insufficient. On the other hand, machine learning assessment requires massive training data that labels human assessment and therefore is expensive. In contrast, prediction-based approaches such as VizML [38] only demand training data describing the dataset and visualizations without the need for human labeling. This reduces the overhead of constructing dataset and thus helps boost the performance. Thus, the above discussion opens up many interesting general questions: How to interpret the prediction-based ML models to understand the reasoning process and potentially derive assessment scores? How to reduce the costs for collecting training datasets?

There exist other research gaps with respect to recommendation. For instance, few ML approaches have yet been applied to recommend insights and non-data-encodings. Is it beneficial to collect training data for those tasks such as predicting important data fields given a data table? Besides, many existing approaches only recommend top-k, separated visualizations given a large data table, while it is unlikely that "top-k visualizations fit all". Future research should study how to recommend dashboards, and even visual analytics systems for more comprehensive and intelligent data analysis. Generating a coherent data story compromising of multiple data facts is another interesting question. Finally, existing systems are confined to well-known charts. Recent research in computer vision has proposed generative models for generating synthetic images. An interesting question is to apply generative models to recommend synthetic, novel visualizations.

6.7 Mining

Mining is an emerging task motivated by the rapid popularization and accumulation of visualization data online (Table 7 and Fig. 12). Generally, there are two kinds of mining tasks, i.e., mining design patterns and mining data patterns, that are discussed in the following text.
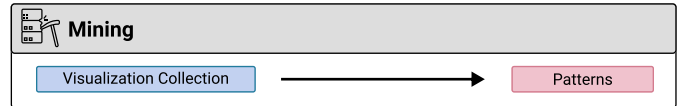

Fig. 12. Input-output model of mining.

Relations to Goals and Other Tasks. The goal of mining is visualization analysis, i.e., to discover useful information or patterns from a visualization collection. Reverse-engineering is often a prerequisite for mining to obtain semantic information.

Relations to Visualization Data. Reasoning is studied on both visualization graphics and programs.

6.7.1 Challenges and Methods

Mining Design Patterns. The concept of design mining refers to leveraging data mining techniques to derive design principles from existing artifacts. Smart *et al.* [95] formally introduced design mining in visualizations and proposed an unsupervised clustering technique to derive common color ramps. However, design mining has been implicitly practiced in several early systems but for different goals. Choudhury and Giles [11] investigate the design patterns (e.g., colored or not) of 300 line graphs sampled from top computer science conferences. Viziometrics [42] computes the average number of visualizations in academic papers from different research domains. In the visualization community, Beagle [40] automatically crawled SVG-based visualizations online to investigate how popular, e.g., line and bar charts are on the web. Hoque and Agrawala [94] performed a design demographics analysis of 7,860 D3 visualizations to identify common patterns such as how frequently circles are used. More recently, Chen *et al.* [8] studied the composition and configuration patterns in multiple-view visualizations collected from visualization papers.

The above systems mostly apply simple statistical analysis [8], [11], [40], [42], [94] or clustering techniques [95]. However, patterns might become meaningless numbers unless they are interpreted as insights or used to empower novel applications, e.g., to motivate design ideas. As such, several systems [8], [94] propose interactive visual interfaces for users to browse the visualization database and find examples. Other approaches also leverage mined design patterns to recommend visual designs [8], [95].

Mining Data Patterns. Another line of research aims to explore the data patterns encoded in visualization ensembles. This concept has been widely implemented in exploring visualization ensembles with respect to a specified type of charts (e.g., ScagExplorer [140] and TimeSeer [141]). We identify two approaches that are irrespective of the chart types, namely Chart Constellations [56] and ChartSeer [9]. Both adopt a visual analytic approach by projecting charts into a 2D space whereby supporting clustering analysis and interactive analysis.

6.7.2 Discussion and Open Questions

There exist several promising directions for future work. For design patterns, existing approaches focus on visualization usage. It would be beneficial to mine underlying semantic patterns from visualization collections, e.g., the

TABLE 8
A Summary of Characteristics of Visualization Data and Research Problems

Characteristics of Visualization Data	Research Problems
It contains multimodal information such as programs, underlying data, and images.	How to represent visualization data [9], translate from one modality to another [31], and fuse and align information between modalities [67].
It is unnatural artifacts purposefully constructed with domain knowledge.	How to automatically infer and/or incorporate with domain knowledge [49].
It is susceptible to detailed local information such as marks' color and size.	How to develop algorithms with higher precision and granularity [31] or involve human in the loop to fix errors made by algorithms [115].
It encodes data and is therefore data-dependent.	How to handle data-dependent problems such as mathematical reasoning [12].

Citations refer to example work instead of full instances.

relationships between linked views in a multiple-view visualization, and the "good" or "bad" practices in existing visualization designs. Those guidelines could motivate the design of recommender systems that automate the creation of visualizations. For instance, many recommender systems in our survey rely on manual coding to derive the design space of visualizations (e.g., [2], [88], [90]). Automated mining of design space would significantly reduce the manual efforts.

Most current mining techniques are limited to simple statistics. However, it is likely that there exist hidden patterns in visualization corpora. Therefore, one could explore the visualization collection with advanced mining techniques and human-in-the-loop analytics to uncover those patterns. A research challenge would be to develop visual analytic systems for analyzing visualization collections.

7 FUTURE RESEARCH OPPORTUNITIES

This survey describes the research vision of formalizing visualizations as a data format. As shown in Table 8, visualization data presents several differences from common data formats such as images and text. Despite many research efforts, there exist sufficient research gaps and potentials for future research. In this section, we outline an organized overview of future research directions.

7.1 Visualization Standards and Interoperability

Our analysis reveals different content formats of visualization data that have been inconsistently adapted (Section 4.1). Such inconsistency impedes interoperability among different visualization systems. Being able to combine different systems and libraries is a common need in application settings. For instance, visualization generation tools such as VizML [38] recommend data encodings, while other systems like MobileVixFixer [2] adjust non-data encodings (visual styles). Both systems do not work well together since

their formats are incompatible with each other. Notably, it is a common practice to select partial information from the full specifications as the intermediate format. This leads to the need for a common standard of visualizations that cover all existing partial formats, as well as derivative tools for auto-completing partial specifications to generate universally compatible formats.

Besides, systems for visualization enhancements usually have to engage in the reverse engineering to extract the underlying programs from graphics. Despite extensive research efforts, reverse engineering remains computationally expensive and lacks robustness [31]. Particularly, our analysis in Section 6.1 suggests that it is currently impossible to perform reverse engineering on bespoke charts. Although recent work like Chartem [7] and VisCode [6] proposes new standards for storing programs in graphics, there is a long way before such standards are adopted in existing systems. Thus, continued research on reverse engineering is essential and beneficial to interoperability.

7.2 Visualization-Tailored Machine Learning

Recent research has increasingly leveraged machine learning to generate or transform visualization. However, it remains challenging to choose the "best" representation and ML models for visualizations [142]. On the one hand, programs are compact and effective representations that are computationally inexpensive [38], [49]. However, programs might not generalize since they are often limited to specific chart types and might not apply to parameter values not observed during training. On the other hand, graphics appear to be a general representation. However, research suggests that off-the-shelf computer vision models for natural images achieve dissatisfactory results in tasks such as visual perceptual learning [66], visual question answering [12], and assessment [61].

This gap is not surprising due to the characteristics of visualization images. Compared with natural images, visualizations are sensitive to local details, e.g., the sizes of graphical marks are of vital importance. Text information is also critical, taking on various roles (e.g., legends, axis labels) that have no tolerance for misinterpretation, e.g., missing a single character in axis labels leads to a great blunder. Those characteristics make it challenging to design an ML model that are tailored to visualizations. Recent work in visualization question answering (e.g., [12], [67], [70]) has proposed sophisticated models to fuse multimodal features and capture visualization-specific features. However, such attempts are limited and their efficacy remains to be thoroughly evaluated.

Another important perspective lies in augmenting machine learning models with knowledge derived from empirical studies. Machine learning models are often criticized for poor explainability. Fortunately, empirical research in the visualization field has accumulated a valuable knowledge base about how visualizations should be interpreted, assessed, and created. It is therefore promising to incorporate that knowledge in ML models that is tailored to visualization research, e.g., Draco [49].

7.3 A Big Data Perspective to Visualization Data

This survey highlights the theme of considering visualizations as a data type. Moving forward, an evolving and

promising theme is big visualization data, which concerns processing and analyzing visualization data at larger scales. This theme leads to several new issues that deserve research efforts. Visualization database systems should effectively store and manage visualizations that are heterogeneous and often unstructured. Moreover, it is unclear how to mine and analyze big visualization data, since existing approaches mainly focus on small-scale visualization collections and use simple statistics. These challenges demand a holistic adaptation of data mining techniques to visualization data, including but not limited to data cleaning, data transformation, data reduction, feature extraction and analysis. For instance, when collecting visualization datasets by web crawling, it is important to clean the visualization collection by removing noise such as non-visualization images. Therefore, there are still many open research topics in terms of analyzing big visualization data.

7.4 Human-Visualization Ecosystem

Although this survey treats visualizations as data processed by AI, we emphasize that visualizations are designed for and arguably central to humans. Therefore, it is important to establish the broader human-visualization ecosystem that AI could engage in and contribute to. As discussed in Section 4, there is a tradeoff between human- and machine-friendliness of visualization data formats that awaits better solutions. It is also helpful to identify the limitations of AI approaches, which offers possibilities of proposing mixed-initiative approaches (e.g., [115]). Besides, it is promising to collect data regarding how humans perceive, assess and use visualizations at scale, which empowers AI to learn new insights and enable learning-based methods for visualization and visual analytics [15]. Finally, as many AI approaches focus on routine tasks (e.g., querying), a critical question arises of how AI could engage humans in creative practices such as creating novel visualizations (e.g., [143]).

8 DISCUSSION AND LIMITATION

In this section, we discuss the limitations in terms of mutual exclusiveness and collective exhaustiveness, as well as generalizability.

8.1 Exclusiveness and Exhaustiveness

In this survey, we use an inductive approach to organize the literature and construct our taxonomy by observing existing work and iteratively generalizing the classification.

We note several dependencies among tasks. For instance, assessment and comparison metrics are often used as optimization functions for recommendation. However, those dependencies should not be interpreted as violations of mutual exclusiveness. Instead, dependencies suggest that tasks can be sequentially combined into a *system pipeline* for solving complex problems.

Because we inductively collected the papers for this survey, we do not claim that our taxonomy is exhaustive. There exist many potential research questions that we have not observed yet. It would be interesting to improve the taxonomy from a deductive perspective by referring to task taxonomies in related fields such as computer vision, artificial intelligence, and databases. For instance, image compression

and style transfer are well-studied tasks in the computer vision community [28], which, however, remain unexplored in the context of visualizations. That said, there exist promising directions for future research of AI4VIS.

8.2 Generalization to Visualizations Beyond Charts

This survey makes simplifying assumptions by focusing on charts and infographics, excluding scientific visualizations and work tailored for a specific type of visualizations. An important concern is how to generalize or extend the taxonomy to a wider spectrum of visualization data. During our analysis process, we find several exceptions that warrant future improvement. In the following text, we discuss notable extensions to our what-why-how taxonomy.

What. Our what taxonomy primarily focuses on visualization data. However, visualizations are hardly considered as standalone data in AI-empowered systems. Instead, it is often necessary to provide ground-truth labels for visualizations as auxiliary data, e.g., chart types [31]. Another type of auxiliary data is user-generated, i.e., interaction logs [144] and analysis provenance [15]. Future research should study the taxonomy of auxiliary data to better contextualize the opportunities for AI for visualization research.

Why. The why taxonomy is organized along two axes: single or many visualizations versus inputting or outputting visualizations. A missing perspective lies in work that neither inputs nor outputs visualizations but instead exploits visualization data in the middle stage. For instance, Lallé *et al.* [145] collected eye-tracking data when browsing visualizations and trained an ML model to predict the learning curves of visualizations. Our survey does not cover this kind of research due to our theme centered on considering visualization as a data format, emphasizing how visualization data are processed and produced. Another perspective for improving our taxonomy is to expand the sub-categories by adding goals that are currently confined to a particular visualization. For instance, deep learning methods for brushing point visualizations [144], [146] or drawing graphs [147] fall under visualization enhancement and generation category, respectively.

How. Echoing the above discussion about "what", there exist corresponding needs of identifying tasks for processing and analyzing auxiliary data. This is crucial since our current task taxonomy concerns visualization data. Moreover, we notice another potential task for visualization data, namely *visualization collection summarization* that aims to represent visualization collections in an effective and compact manner. However, existing approaches are limited and specific to visualization types, e.g., Scagnostics for scatterplots [140] and line charts [141]. In addition to those statistical summarization approaches, more recent visual analytic approaches use glyphs [9]. Future work could provide a systematical overview of visualization summarization with increasing research efforts.

9 CONCLUSION

This survey probes the concept of considering visualizations as an emerging data format and investigate the advance of applying artificial intelligence to visualization

data (AI4VIS). We present a novel classification that enables the readers to find relevant literature among a wide variety of research areas. Our classification can also help readers to understand current techniques and find areas for future research. We hope that our survey could help stimulate new theories, problems, techniques, and applications.

ACKNOWLEDGMENTS

REFERENCES

[1] C. Qian, S. Sun, W. Cui, J.-G. Lou, H. Zhang, and D. Zhang, "Retrieve-then-adapt: Example-based automatic generation for proportion-related infographics," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 443–452, Feb. 2021.

[2] A. Wu, W. Tong, T. Dwyer, B. Lee, P. Isenberg, and H. Qu, "MobileVisFixer: Tailoring web visualizations for mobile phones leveraging an explainable reinforcement learning framework," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 464–474, Feb. 2021.

[3] M. Oppermann, R. Kincaid, and T. Munzner, "VizCommender: Computing text-based similarity in visualization repositories for content-based recommendations," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 495–505, Feb. 2021.

[4] D. Shi, X. Xu, F. Sun, Y. Shi, and N. Cao, "Calliope: Automatic visual data story generation from a spreadsheet," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 453–463, Feb. 2021.

[5] Y. Kim and J. Heer, "Gemini: A grammar and recommender system for animated transitions in statistical graphics," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 485–494, Feb. 2021.

[6] P. Zhang, C. Li, and C. Wang, "VisCode: Embedding information in visualization images using encoder-decoder network," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 326–336, Feb. 2021.

[7] J. Fu *et al.*, "Chartem: Reviving chart images with data embedding," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 337–346, Feb. 2021.

[8] X. Chen, W. Zeng, Y. Lin, H. M. Al-Maneea , J. Roberts, and R. Chang, "Composition and configuration patterns in multiple-view visualizations," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 1514–1524, Feb. 2021.

[9] J. Zhao, M. Fan, and M. Feng, "ChartSeer: Interactive steering exploratory visual analysis with machine intelligence," *IEEE Trans. Vis. Comput. Graph.*, to be published, doi: 10.1109/TVCG.2020.3018724.

[10] Z. Chen, M. Cafarella, and E. Adar, "DiagramFlyer: A search engine for data-driven diagrams," in *Proc. Int. Conf. World Wide Web*, 2015, pp. 183–186.

[11] S. Ray Choudhury and C. L. Giles, "An architecture for information extraction from figures in digital libraries," in *Proc. Int. Conf. World Wide Web*, 2015, pp. 667–672.

[12] K. Kafle, B. Price, S. Cohen, and C. Kanan, "DVQA: Understanding data visualizations via question answering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5648–5656.

[13] C. Chen, R. Zhang, E. Koh, S. Kim, S. Cohen, and R. Rossi, "Figure captioning with relation maps for reasoning," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2020, pp. 1537–1545.

[14] L. McNabb and R. S. Laramee, "Survey of surveys (SOS)-mapping the landscape of survey papers in information visualization," *Comput. Graph. Forum*, vol. 36, no. 3, pp. 589–617, 2017.

[15] K. Xu, A. Ottley, C. Walchshofer, M. Streit, R. Chang, and J. Wenskovitch, "Survey on the analysis of user interactions and visualization provenance," *Comput. Graph. Forum*, vol. 39, no. 3, pp. 757–783, 2020.

[16] J. P. Ono, R. S. Hong, C. T. Silva, and J. Freire . " Why should we teach machines to read charts made for humans?," 2018. [Online]. Available: https://vgc.poly.edu/ jhenrique/files/ chi2019_workshop_ML_Evaluate_Vis.pdf

[17] B. Lee *et al.*, "Broadening intellectual diversity in visualization research papers," *IEEE Comput. Graph. Appl.*, vol. 39, no. 4, pp. 78–85, Jul. /Aug. 2019.

[18] Visualization meets AI workshop. Accessed: Nov. 12, 2020. [Online]. Available: https://visai2020.github.io/

[19] G. Andrienko *et al.*, "Big data visualization and analytics: Future research challenges and emerging applications," in *Proc. BigVis: Big Data Vis. Exploration Analytics*, 2020, pp. 1–8.

[20] B. Saket, D. Moritz, H. Lin, V. Dibia, C. Demiralp, and J. Heer, "Beyond heuristics: Learning visualization design," 2018, *arXiv:1807.06641*.

[21] S. Zhu, G. Sun, Q. Jiang, M. Zha, and R. Liang, "A survey on automatic infographics and visualization recommendations," *Vis. Inform.*, vol. 4, no. 3, pp. 24–40, 2020.

[22] X. Qin, Y. Luo, N. Tang, and G. Li, "Making data visualization more efficient and effective: A survey," *VLDB J.*, vol. 29, no. 1, pp. 93–117, 2020.

[23] K. Davila, S. Setlur, D. Doermann, U. K. Bhargava, and V. Govindaraju, "Chart mining: A survey of methods for automated chart analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published, doi: 10.1109/TPAMI.2020.2992028.

[24] Q. Wang, Z. Chen, Y. Wang, and H. Qu, "Applying machine learning advances to data visualization: A survey on ML4VIS," 2020, *arXiv:2012.00467*.

[25] J. Kehrer and H. Hauser, "Visualization and visual analysis of multifaceted scientific data: A survey," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 3, pp. 495–513, Mar. 2013.

[26] M. Behrisch *et al.*, "Quality metrics for information visualization," in *Comput. Graph. Forum*, vol. 37, no. 3. pp. 625–662, 2018.

[27] A. Srinivasan and J. Stasko, "Natural language interfaces for data analysis with visualization: Considering what has and could be asked," in *Proc. Eurograph./IEEE VGTC Conf. Vis.*, 2017, pp. 55–59.

[28] Computer vision | paper with codes. Accessed: Nov. 12, 2020. [Online]. Available: https://paperswithcode.com/area/computer-vision

[29] Y. Luo, C. Chai, X. Qin, N. Tang, and G. Li, "Interactive cleaning for progressive visualization through composite questions," in *Proc. IEEE Int. Conf. Data Eng. .* 2020, pp. 733–744.

[30] A. Satyanarayan *et al.*, "Critical reflections on visualization authoring systems," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 461–471, 2010.

[31] J. Poco and J. Heer, "Reverse-engineering visualizations: Recovering visual encodings from chart images," *Comput.Graph. Forum*, vol. 36, no. 3, pp. 353–363, 2017.

[32] M. Savva, N. Kong, A. Chhajta, L. Fei-Fei , M. Agrawala, and J. Heer, "Revision: Automated classification, analysis and redesign of chart images," in *Proc. Annu. ACM Symp. User Interface Softw. Technol.*, 2011, pp. 393–402.

[33] D. Moritz, "Text detection in screen images with a convolutional neural network," *J. Open Source Softw.*, vol. 2, no. 15, Jul. 2017, Art. no. 235. [Online]. Available: https://doi.org/10.21105/joss.00235

[34] J. Harper and M. Agrawala, "Deconstructing and restyling D3 visualizations," in *Proc. ACM Symp. User Interface Softw. Technol.*, 2014, pp. 253–262.

[35] M. Bostock, V. Ogievetsky, and J. Heer, "D³ data-driven documents," *IEEE Trans. Vis. Comput. Graph.*, vol. 17, no. 12, pp. 2301–2309, Dec. 2011.

[36] F. Bolte and S. Bruckner, "Vis-a-Vis: Visual exploration of visualization source code evolution," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 7, pp. 3153–3167, Jul. 2021.

[37] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer, "Vega-Lite: A grammar of interactive graphics," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 1, pp. 341–350, Jan. 2017.

[38] K. Hu, M. A. Bakker, S. Li, T. Kraska, and C. Hidalgo, "VizML: A machine learning approach to visualization recommendation," in *Proc. Conf. Hum. Factors Comput. Syst.*, 2019, pp. 1–12.

[39] M. Raji, J. Duncan, T. Hobson, and J. Huang, "Dataless sharing of interactive visualization," *IEEE Trans. Vis. Comput. Graph.*, to be published, doi: 10.1109/TVCG.2020.2984708.

[40] L. Battle, P. Duan, Z. Miranda, D. Mukusheva, R. Chang, and M. Stonebraker, "Beagle: Automated extraction and interpretation of visualizations from the web," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2018, pp. 1–8.

[41] M. A. Borkin *et al.* "What makes a visualization memorable?," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 12, pp. 2306–2315, Dec. 2013.

[42] P.-s. Lee, J. D. West, and B. Howe, "Viziometrics: Analyzing visual information in the scientific literature," *IEEE Trans. Big Data*, vol. 4, no. 1, pp. 117–129, Mar. 2018.

[43] K. Hu *et al.*, "VizNet: Towards a large-scale visualization learning and benchmarking repository," in *Proc. ACM Conf. Hum Factors Comput. Syst.*, 2019, pp. 1–12.

[44] A. Rule, A. Tabard, and J. D. Hollan, "Exploration and explanation in computational notebooks," in *Proc. Conf. Hum. Factors Comput. Syst.* 2018, pp. 1–12.

[45] S. R. Choudhury, S. Wang, and C. L. Giles, "Scalable algorithms for scholarly figure mining and semantics," in *Proc. Int. Workshop Semantic Big Data*, 2016, pp. 1–6.

[46] J. Choi, S. Jung, D. G. Park, J. Choo, and N. Elmqvist, "Visualizing for the non-visual: Enabling the visually impaired to use visualization," *Comput. Graph. Forum*, vol. 38, no. 3, pp. 249–260, 2019.

[47] W. Huang and C. L. Tan, "A system for understanding imaged infographics and its applications," in *Proc. ACM Symp. Document Eng.*, 2007, pp. 9–18.

[48] R. Burns, S. Carberry, S. Elzer, and D. Chester, "Automatically recognizing intended messages in grouped bar charts," in *Proc. Int. Conf. Theory App. Diagrams*, 2012, pp. 8–22.

[49] D. Moritz *et al.* "Formalizing visualization design knowledge as constraints: Actionable and extensible models in draco," *IEEE Trans. Vis. Comput. Graph.* vol. 25, no. 1, pp. 438–448, Jan. 2019.

[50] H. Lin, D. Moritz, and J. Heer, "Dziban: Balancing agency & automation in visualization design via anchored recommendations," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2020, pp. 1–12.

[51] E. Kim and K. F. McCoy, "Multimodal deep learning using images and text for information graphic classification," in *Proc. ACM Int. Conf. Comput. Accessib.*, 2018, pp. 143–148.

[52] Y. Luo, X. Qin, C. Chai, N. Tang, G. Li, and W. Li, "Steerable self-driving data visualization," *IEEE Trans. Knowl. Data Eng.*, to be published, doi: 10.1109/TKDE.2020.2981464.

[53] Y. Luo, X. Qin, N. Tang, and G. Li, "DeepEye: Towards automatic data visualization," in *Proc. Int. Conf. Data Eng.*, 2018, pp. 101–112.

[54] A. Key, B. Howe, D. Perry, and C. Aragon, "VizDeck: self-organizing dashboards for visual analytics," in *Proc. ACM Int. Conf. Manage. Data*, 2012, pp. 681–684.

[55] Y. Luo, X. Qin, N. Tang, G. Li, and X. Wang, "DeepEye: Creating good data visualizations by keyword search," in *Proc. Int. Conf. Manag. Data*, 2018, pp. 1733–1736.

[56] S. Xu, C. Bryan, J. K. Li, J. Zhao, and K.-L. Ma, "Chart constellations: Effective chart summarization for collaborative and multi-user analyses," *Comput. Graph. Forum*, vol. 37, no. 3, 2018, pp. 75–86.

[57] Z. Bylinskii *et al.*, "Learning visual importance for graphic designs and data visualizations," in *Proc. ACM Symp. User Interface Softw. Technol.*, 2017, pp. 57–69.

[58] A. Y. Lin, J. Ford, E. Adar, and B. Hecht, "VizByWiki: Mining data visualizations from the web to enrich news articles," in *Proc. World Wide Web Conf.*, 2018, pp. 873–882.

[59] R. Ma *et al.*, "LADV: Deep learning assisted authoring of dashboard visualizations from images and sketches," *IEEE Trans. Vis. Comput. Graph.*, to be published, doi: 10.1109/TVCG.2020.2980227.

[60] N. Siegel, Z. Horvitz, R. Levin, S. Divvala, and A. Farhadi, "FigureSeer: Parsing result-figures in research papers," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 664–680.

[61] X. Fu, Y. Wang, H. Dong, W. Cui, and H. Zhang, "Visualization assessment: A machine learning approach," in *Proc. IEEE Vis. Conf.,*, 2019, pp. 126–130.

[62] W. Dai, M. Wang, Z. Niu, and J. Zhang, "Chart decoder: Generating textual and numeric information from chart images automatically," *J. Vis. Lang. Comput.*, vol. 48, pp. 101–109, 2018.

[63] P. Chagas *et al.*, "Evaluation of convolutional neural network architectures for chart image classification," in *Proc. Int. Joint Conf. Neural Netw.*, 2018, pp. 1–8.

[64] S. Tsutsui and D. J. Crandall, "A data driven approach for compound figure separation using convolutional neural networks," in *Proc. Int. Conf. Document Anal. Recognit.*, 2017, vol. 1, pp. 533–540.

[65] B. Tang *et al.*, "DeepChart: Combining deep convolutional networks and deep belief networks in chart classification," *Signal Processing*, vol. 124, pp. 156–161, 2016.

[66] D. Haehn, J. Tompkin, and H. Pfister, "Evaluating graphical perception with CNNs," *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 1, pp. 641–650, Jan. 2019.

[67] R. Chaudhry, S. Shekhar, U. Gupta, P. Maneriker, P. Bansal, and A. Joshi, "Leaf-QA: Locate, encode & attend for figure question answering," in *IEEE Winter Conf. Appl. Comput. Vis.*, 2020, pp. 3512–3521.

[68] C. Chen, R. Zhang, S. Kim, S. Cohen, T. Yu, R. Rossi, and R. Bunescu, "Neural caption generation over figures," in *Adjunct Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput. Proc. ACM Int. Symp. Wearable Comput.*, 2019, pp. 482–485.

[69] S. E. Kahou, V. Michalski, A. Atkinson, Á. Kádár, A. Trischler, and Y. Bengio, "Figureqa: An annotated figure dataset for visual reasoning," 2017, *arXiv:1710.07300*.

[70] N. Methani, P. Ganguly, M. M. Khapra, and P. Kumar, "PlotQA: Reasoning over scientific plots," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2020, pp. 1527–1536.

[71] R. Reddy, R. Ramesh, A. Deshpande, and M. M. Khapra, "Figurenet: A deep learning model for question-answering on scientific plots," in *Proc. Int. Joint Conf. Neural Netw.*, 2019, pp. 1–8.

[72] H. Singh and S. Shekhar, "STL-CQA: Structure-based transformers with localization and encoding for chart question answering," in *Proc. Conf. Empirical Methods Nat. Lang. Process.*, 2020, pp. 3275–3284.

[73] J. Obeid and E. Hoque, "Chart-to-text: Generating natural language descriptions for charts by adapting the transformer model," in *Proc. Int. Conf. Nat. Lang. Gener.*, 2020, pp. 138–147.

[74] V. Dibia and Ç. Demiralp, "Data2Vis: Automatic generation of data visualizations using sequence-to-sequence recurrent neural networks," *IEEE Comput. Graph. Appl.*, vol. 39, no. 5, pp. 33–46, Sep. /Oct. 2019.

[75] C. Wang, Y. Feng, R. Bodik, A. Cheung, and I. Dillig, "Visualization by example," *Proc. ACM Program. Lang.*, vol. 4, no. 49, pp. 1–28, 2019.

[76] V. Lifschitz, "Action languages, answer sets, and planning," in *The Logic Programming Paradigm*. Berlin, Germany: Springer, 1999, pp. 357–373.

[77] H. Ehsan, M. A. Sharaf, and P. K. Chrysanthis, "Efficient recommendation of aggregate data visualizations," *IEEE Tran. Knowl. Data Eng.*, vol. 30, no. 2, pp. 263–277, Feb. 2018.

[78] H. Ehsan, M. A. Sharaf, and P. K. Chrysanthis, "MuVE: Efficient multi-objective view recommendation for visual data exploration," in *Proc. Int. Conf. Data Eng.*, 2016, pp. 731–742.

[79] R. Mafrur, M. A. Sharaf, and H. A. Khan, "Dive: Diversifying view recommendation for visual data exploration," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2018, pp. 1123–1132.

[80] T. Siddiqui, A. Kim, J. Lee, K. Karahalios, and A. Parameswaran, "Effortless data exploration with zenvisage: An expressive and interactive visual analytics system," *Proc. VLDB Endowment*, vol. 10, no. 4, pp. 126–130, 2016.

[81] M. Vartak, S. Rahman, S. Madden, A. Parameswaran, and N. Polyzotis, "SeeDb: Efficient data-driven visualization recommendations to support visual analytics," in *Proc. VLDB Endowment Int. Conf. Very Large Data Bases*, vol. 8, no. 13, 2015, Art. no. 2182.

[82] E. Wu *et al.*, "Combining design and performance in a data visualization management system," in *Proc. Biennial Conf. Innov. Data Syst. Res.*, 2017, pp. 1–11.

[83] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer, "Towards a general-purpose query language for visualization recommendation," in *Proc. Workshop Hum.-In-the-Loop Data Analytics*, 2016, pp. 1–6.

[84] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer, "Voyager: Exploratory analysis via faceted browsing of visualization recommendations," *IEEE Trans. Vis. Comput. Graph.*, vol. 22, no. 1, pp. 649–658, Jan. 2016.

[85] K. Wongsuphasawat *et al.* "Voyager 2: Augmenting visual analysis with partial view specifications," in *Proc. of the ACM Conf. Hum. Factors Comput. Syst.*, 2017, pp. 2648–2659.

[86] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[87] J. Mackinlay, "Automating the design of graphical presentations of relational information," *ACM Trans. Graph.*, vol. 5, no. 2, pp. 110–141, 1986.

[88] Y. Wang *et al.*, "DataShot: Automatic generation of fact sheets from tabular data," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 895–905, Jan. 2020.

[89] J. Harper and M. Agrawala, "Converting basic D3 charts into reusable style templates," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 3, pp. 1274–1286, Mar. 2018.

[90] W. Cui *et al.*, "Text-to-Viz: Automatic generation of infographics from proportion-related natural language statements," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 906–916, Jan. 2020.

[91] C. Lai, Z. Lin, R. Jiang, Y. Han, C. Liu, and X. Yuan, "Automatic annotation synchronizing with textual description for visualization," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2020, pp. 1–13.

[92] N. Kong and M. Agrawala, "Graphical overlays: Using layered elements to aid chart reading," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 12, pp. 2631–2638, Dec. 2012.

[93] B. Saleh, M. Dontcheva, A. Hertzmann, and Z. Liu, "Learning style similarity for searching infographics," in *Proc. Graph. Interface Conf.*, 2015, pp. 59–64.

[94] E. Hoque and M. Agrawala, "Searching the visual style and structure of D3 visualizations," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 1236–1245, Jan. 2020.

[95] S. Smart, K. Wu, and D. A. Szafir, "Color crafting: Automating the construction of designer quality color ramps," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 1215–1225, Jan. 2020.

[96] D. H. Kim, E. Hoque, and M. Agrawala, "Answering questions about charts and generating visual explanations," in *Proc. ACM Conf Hum Factors Comput. Syst.*, 2020, pp. 1–13.

[97] J. Brosz, M. A. Nacenta, R. Pusch, S. Carpendale, and C. Hurter, "Transmogrification: Causal manipulation of visualizations," in *Proc. ACM Symp. User Interface Softw. Technol.*, 2013, pp. 97–106.

[98] J. E. Zhang, N. Sultanum, A. Bezerianos, and F. Chevalier, "Dataquilt: Extracting visual elements from images to craft pictorial visualizations," in *Proc. Conf. Hum. Factors Comput. Syst.*, 2020, pp. 1–13.

[99] Y. Zhou and C. L. Tan, "Learning-based scientific chart recognition," in *Proc. of the IAPR Int. Workshop Graph. Recognit.*, 2001, pp. 482–492.

[100] Q. Wang, Z. Li, S. Fu, W. Cui, and H. Qu, "Narvis: Authoring narrative slideshows for introducing data visualization designs," *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 1, pp. 779–788, Jan. 2019.

[101] M. Lu, J. Liang, Y. Zhang, G. Li, S. Chen, Z. Li, and X. Yuan, "Interaction+: Interaction enhancement for web-based visualizations," in *Proc. IEEE Pacific Vis. Symp.*, 2017, pp. 61–70.

[102] W. Huang, R. Liu, and C.-L. Tan, "Extraction of vectorized graphical information from scientific chart images," in *Proc. Int. Conf. Document Anal. Recognit.*, 2007, vol. 1, pp. 521–525.

[103] J. Poco, A. Mayhua, and J. Heer, "Extracting and retargeting color mappings from bitmap images of visualizations," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 1, pp. 637–646, Jan. 2018.

[104] W. Browuer, S. Kataria, S. Das, P. Mitra, and C. L. Giles, "Segregating and extracting overlapping data points in two-dimensional plots," in *Proc. ACM/IEEE-CS Joint Conf. Digit. Libraries*, 2008, pp. 276–279.

[105] S. Ray Choudhury, S. Wang, and C. L. Giles, "Curve separation for line graphs in scholarly documents," in *Proc. ACM/IEEE-CS Joint Conf. Digit. Libraries*, 2016, pp. 277–278.

[106] C. Bryan, K.-L. Ma, and J. Woodring, "Temporal summary images: An approach to narrative visualization via interactive annotation generation and placement," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 1, pp. 511–520, Jan. 2017.

[107] R. Savvides, A. Henelius, E. Oikarinen, and K. Puolamäki, "Significance of patterns in data visualisations," in *Proc. ACM Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 1509–1517.

[108] D. J.-L. Lee, H. Dev, H. Hu, H. Elmeleegy, and A. Parameswaran, "Avoiding drill-down fallacies with vispilot: Assisted exploration of data subsets," in *Proc. Int. Conf. Intell. User Interfaces*, 2019, pp. 186–196.

[109] J. Gao, Y. Zhou, and K. E. Barner, "View: Visual information extraction widget for improving chart images accessibility," in *Proc. Int. Conf. Image Process.*, 2012, pp. 2865–2868.

[110] Z. Chen, Y. Wang, Q. Wang, Y. Wang, and H. Qu, "Towards automated infographic design: Deep learning-based auto-extraction of extensible timeline," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 917–926, Jan. 2020.

[111] R. A. Al-Zaidy and C. L. Giles, "A machine learning approach for semantic structuring of scientific charts in scholarly documents." in *Proc. Conf. Artif. Intell.*, 2017, pp. 4644–4649.

[112] V. S. N. Prasad, B. Siddiquie, J. Golbeck, and L. S. Davis, "Classifying computer generated charts," in *Proc. Int. Workshop Content Based Multimedia Indexing*, 2007, pp. 85–92.

[113] R. A. Al-Zaidy, S. R. Choudhury, and C. L. Giles, "Automatic summary generation for scientific data charts," in *Proc. Conf. Artif. Intell.*, 2016, pp. 658–663.

[114] G. G. Méndez, M. A. Nacenta, and S. Vandenheste, "iVoLVER: Interactive visual language for visualization extraction and reconstruction," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2016, pp. 4073–4085.

[115] D. Jung *et al.*, "ChartSense: Interactive data extraction from chart images," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2017, pp. 6706–6717.

[116] D. McCandless, Information is Beautiful. Scotland,Scotland, UK: Collins, 2009.

[117] S. Kandel, R. Parikh, A. Paepcke, J. M. Hellerstein, and J. Heer, "Profiler: Integrated statistical analysis and visualization for data quality assessment," in *Proc. Int. Working Conf. Adv. Visual Interfaces*, 2012, pp. 547–554.

[118] P.-M. Law, R. C. Basole, and Y. Wu, "Duet: Helping data analysis novices conduct pairwise comparisons by minimal specification," *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 1, pp. 427–437, Jan. 2018.

[119] Y. Kim, K. Wongsuphasawat, J. Hullman, and J. Heer, "GraphScape: A model for automated reasoning about visualization similarity and sequencing," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2017, pp. 2628–2638.

[120] Y. Ma, A. K. Tung, W. Wang, X. Gao, Z. Pan, and W. Chen, "ScatterNet: A deep subjective similarity model for visual analysis of scatterplots," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 3, pp. 1562–1576, Mar. 2018.

[121] L. Cerulo and G. Canfora, "A taxonomy of information retrieval models and tools," *J. Comput. Inf. Technol.*, vol. 12, no. 3, pp. 175–194, 2004.

[122] A. Srinivasan, S. M. Drucker, A. Endert, and J. Stasko, "Augmenting visualizations with interactive data facts to facilitate interpretation and communication," *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 1, pp. 672–681, Jan. 2018.

[123] Z. Li, S. Carberry, H. Fang, K. F. McCoy, and K. Peterson, "Infographics retrieval: A new methodology," in *Proc. Int. Conf. Appl. Nat. Lang. Data Bases Inf. Syst.*, 2014, pp. 101–113.

[124] Z. Li, S. Carberry, H. Fang, K. F. McCoy, K. Peterson, and M. Stagitis, "A novel methodology for retrieving infographics utilizing structure and message content," *Data Knowl. Eng.*, vol. 100, pp. 191–210, 2015.

[125] M. Wattenberg, "Sketching a graph to query a time-series database," in *Proc. Extended Abstr. Hum. Factors Comput. Syst.*, 2001, pp. 381–382.

[126] C. Fan, K. Matkovic, and H. Hauser, "Sketch-based fast and accurate querying of time series using parameter-sharing LSTM networks," *IEEE Trans. Vis. Comput. Graph.*, to be published, doi: 10.1109/TVCG.2020.3002950.

[127] V. O. Mittal, J. D. Moore, G. Carenini, and S. Roth, "Describing complex charts in natural language: A caption generation system," *Comput. Linguistics*, vol. 24, no. 3, pp. 431–467, 1998.

[128] Z. Cui, S. K. Badam, M. A. Yalçin, and N. Elmqvist, "DataSite: Proactive visual data exploration with computation of insight-based recommendations," *Inf. Vis.*, vol. 18, no. 2, pp. 251–267, 2019.

[129] S. Demir, S. Carberry, and K. F. McCoy, "Summarizing information graphics textually," *Comput. Linguistics*, vol. 38, no. 3, pp. 527–574, 2012.

[130] C. Liu, L. Xie, Y. Han, D. Wei, and X. Yuan, "AutoCaption: An approach to generate natural language description from visualization automatically," in *Proc. IEEE Pacific Vis. Symp.*, 2020, pp. 191–195.

[131] W. S. Cleveland and R. McGill, "Graphical perception: Theory, experimentation, and application to the development of graphical methods," *J. Amer. Stat. Assoc.*, vol. 79, no. 387, pp. 531–554, 1984.

[132] C. Demiralp, P. J. Haas, S. Parthasarathy, and T. Pedapati, "Foresight: Recommending visual insights," *Proc. VLDB Endowment*, vol. 10, no. 12, 2017.

[133] R. Ding, S. Han, Y. Xu, H. Zhang, and D. Zhang, "Quickinsights: Quick and automatic discovery of insights from multi-dimensional data," in *Proc. Int. Conf. Manag. Data*, 2019, pp. 317–332.

[134] J. Mackinlay, P. Hanrahan, and C. Stolte, "Show me: Automatic presentation for visual analysis," *IEEE Trans. Vis. Comput. Graph.*, vol. 13, no. 6, pp. 1137–1144, Nov./Dec. 2007.

[135] R. Ananthanarayanan, P. K. Lohia, and S. Bedathur, "Datavizard: Recommending visual presentations for structured data," in *Proc. Int. Workshop Web Databases*, 2018, pp. 1–6.

[136] A. Narechania, A. Srinivasan, and J. Stasko, "NL4DV: A toolkit for generating analytic specifications for data visualization from natural language queries," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 369–379, Feb. 2021.

[137] F. Bouali, A. Guettala, and G. Venturini, "VizAssist: An interactive user assistant for visual data mining," *Vis. Comput.*, vol. 32, no. 11, pp. 1447–1463, 2016.

[138] Z. Chen, W. Tong, Q. Wang, B. Bach, and H. Qu, "Augmenting static visualizations with PapARVis designer," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2020, pp. 1–12.

[139] B. Yu and C. T. Silva, "FlowSense: A natural language interface for visual data exploration within a dataflow system," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 1–11, Jan. 2020.

[140] T. N. Dang and L. Wilkinson, "ScagExplorer: Exploring scatterplots by their scagnostics," in *Proc. IEEE Pacific Vis. Symp.*, 2014, pp. 73–80.

[141] T. N. Dang, A. Anand, and L. Wilkinson, "TimeSeer: Scagnostics for high-dimensional time series," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 3, pp. 470–483, Mar. 2013.

[142] A. Wu, L. Xie, B. Lee, Y. Wang, W. Cui, and H. Qu, "Learning to automate chart layout configurations using crowdsourced paired comparison," 2021, *arXiv:2101.03680*.

[143] L.-P. Yuan, Z. Zhou, J. Zhao, Y. Guo, F. Du, and H. Qu, "InfoColorizer: Interactive recommendation of color palettes for infographics," 2021, *arXiv:2102.02041*.

[144] C. Fan and H. Hauser, "Fast and accurate CNN-based brushing in scatterplots," *Comput. Graph. Forum*, vol. 37, no. 3, pp. 111–120, 2018.

[145] S. Lallé, D. Toker, C. Conati, and G. Carenini, "Prediction of users' learning curves for adaptation while using an information visualization," in *Proc. Int. Conf. Intell. User Interfaces*, 2015, pp. 357–368.

[146] Z. Chen, W. Zeng, Z. Yang, L. Yu, C.-W. Fu, and H. Qu, "LassoNet: Deep lasso-selection of 3D point clouds," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 195–204, Jan. 2020.

[147] Y. Wang, Z. Jin, Q. Wang, W. Cui, T. Ma, and H. Qu, "DeepDrawing: A deep learning approach to graph drawing," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 676–686, Jan. 2020.
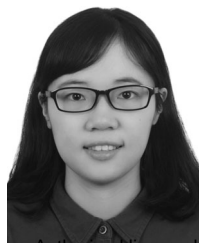
Aoyu Wu received the BE and ME degrees from the Hong Kong University of Science and Technology (HKUST). He is currently working toward the PhD degree with the Department of Computer Science and Engineering, HKUST. His research interests include data visualization and human-computer interaction.

Yun Wang received the BEng degree from Fudan University and the PhD degree in computer science and engineering from the Hong Kong University of Science and Technology. She is currently a senior researcher with Microsoft Research. Her research interests include data visualization and human-computer interaction, with the focus on facilitating data storytelling, data-driven design, and human-AI collaboration. She has authored or coauthored papers extensively in *IEEE Visualization*, the *IEEE Transactions on Visualization and Computer Graphics*, ACM Conference on Human Factors in Computing Systems.

Xinhuan Shu received the BE degree in computer science and technology from Zhejiang University, China, in 2017. She is currently working toward the PhD degree with the Department of Computer Science and Engineering, the Hong Kong University of Science and Technology. Her research interests include data-driven storytelling, animated visualization, and visual analytics.

Dominik Moritz received the PhD degree in computer science from the University of Washington. He is currently a professor with HumanComputer Interaction Institute, Carnegie Mellon University (CMU) and a researcher with Apple. At CMU, he coleads the Data Interaction Group. He enhances people's ability to understand and communicate large and complex data with methods that integrate the capabilities of both people and machines.

Weiwei Cui received the BS degree in computer science and technology from Tsinghua University, China, and the PhD degree in computer science and engineering from the Hong Kong University of Science and Technology, Hong Kong. He is currently a principal researcher with Microsoft Research Asia, China. His current research focuses on visualization, with the focuses on democratizing visualization and AI-assisted design.

Haidong Zhang received the PhD degree in computer science from Peking University, China. He is currently a principal architect with Microsoft Research Asia. His research interests include visualization and human-computer interaction.

Dongmei Zhang received the BE and ME degrees from Tsinghua University, and the PhD degree in robotics from the School of Computer Science, Carnegie Mellon University. She is currently a distinguished scientist and an assistant managing director with Microsoft Research Asia. Her research interests include data, knowledge and intelligence with research directions in data intelligence, knowledge computing, information visualization, and software engineering.

Huamin Qu received the BS degree in mathematics from Xi'an Jiaotong University, China, and the MS and PhD degrees in computer science from the Stony Brook University. He is currently a professor with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology (HKUST) and the director of the Interdisciplinary Program Office, HKUST. His research interests mainly include visualization and human-computer interaction, with focuses on urban informatics, social network analysis, E-learning, text visualization, and explainable artificial intelligence.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.