TU Delft

Delft University of Technology

ChartStory

Automated Partitioning, Layout, and Captioning of Charts into Comic-Style Narratives

Zhao, Jian; Xu, Shenyu; Chandrasegaran, Senthil; Bryan, Christopher James; Du, Fan; Mishra, Aditi; Qian, Xin; Li, Yiran; Ma, Kwan Liu

DOI
10.1109/TVCG.2021.3114211

Publication date
2022

Document Version
Final published version

Published in
IEEE Transactions on Visualization and Computer Graphics

Citation (APA)
Zhao, J., Xu, S., Chandrasegaran, S., Bryan, C. J., Du, F., Mishra, A., Qian, X., Li, Y., & Ma, K. L. (2022). ChartStory: Automated Partitioning, Layout, and Captioning of Charts into Comic-Style Narratives. *IEEE Transactions on Visualization and Computer Graphics*, *29*(2), 1384-1399. https://doi.org/10.1109/TVCG.2021.3114211

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Green Open Access added to TU Delft Institutional Repository

'You share, we take care!' - Taverne project

https://www.openaccess.nl/en/you-share-we-take-care

Otherwise as indicated in the copyright section: the publisher is the copyright holder of this work and the author uses the Dutch legislation to make this work public.

ChartStory: Automated Partitioning, Layout, and Captioning of Charts into Comic-Style Narratives

Jian Zhao, Shenyu Xu, Senthil Chandrasegaran, Chris Bryan, Fan Du, Aditi Mishra, Xin Qian, Yiran Li, and Kwan-Liu Ma, *Fellow, IEEE*

Abstract—Visual data storytelling is gaining importance as a means of presenting data-driven information or analysis results, especially to the general public. This has resulted in design principles being proposed for data-driven storytelling, and new authoring tools being created to aid such storytelling. However, data analysts typically lack sufficient background in design and storytelling to make effective use of these principles and authoring tools. To assist this process, we present *ChartStory* for crafting data stories from a collection of user-created charts, using a style akin to comic panels to imply the underlying sequence and logic of data-driven narratives. Our approach is to operationalize established design principles into an advanced pipeline that characterizes charts by their properties and similarities to each other, and recommends ways to partition, layout, and caption story pieces to serve a narrative. ChartStory also augments this pipeline with intuitive user interactions for visual refinement of generated data comics. We extensively and holistically evaluate ChartStory via a trio of studies. We first assess how the tool supports data comic *creation* in comparison to a manual baseline tool. Data comics from this study are subsequently compared and evaluated to ChartStory's automated recommendations by a team of narrative visualization practitioners. This is followed by a pair of interview studies with data scientists using their own datasets and charts who provide an additional assessment of the system. We find that ChartStory provides cogent recommendations for narrative generation, resulting in data comics that compare favorably to manually-created ones.

Index Terms—Data story generation, narrative visualization, data-driven storytelling, data comics

1 Introduction

Visual data storytelling concerns the communication of data insights and narratives to general audiences using engaging visualizations. The notion of "visual data story" [28] includes the discovery of interesting information in data as "story pieces," the representation of them using visualizations and annotations, and the sequencing of these representations into a narrative to communicate a high-level goal. When designed well, visual stories can greatly improve the comprehension of data even among laypeople.

However, the creation of visual data stories can be especially difficult for a general data analyst, who is familiar with the data, but often has little background in art or design. Research on computational notebooks highlights the tension between *exploration*—conducted at a personal scale with messy results—and *explanation*—where results are cleaned up at the expense of provenance [23], [36], [45]. There is a need for an intermediate step to mitigate this tension. One main challenge is identifying a narrative—converting the analysis results (e.g., a collection of charts) into a compelling story that resonates with the reader and reveals a logical progression of ideas. The second is presenting the narrative—arranging the charts and textual annotations by leveraging design principles drawn from research on visual data storytelling.

There exist theoretical frameworks to generate a narrative from a set of charts [22], [28]. Yet, no satisfactory approach has been proposed to operationalize these theories. Interactive authoring tools for presenting a narrative in through expressive visualizations [38], [54], annotations [35], and comic-style storyboards [24] also exist, but require much manual input and learning, and lack support for automated narrative identification or presentation. In contrast, several works focus on automatically generating specific types of visual data stories such as "fact sheets" directly from data tables [13], [41], [49]. However, they are mainly designed for a general audience, not data analysts who wish to fully control the analysis and only automate the narrative generation. These tools also focus on automated grouping and sequencing of charts based on similarity measures and edit distances, but seldom incorporate theoretical considerations of general narrative such as story pieces [22] and comic-style narrative [8] (see Fig. 2). ChartStory operationalizes these theoretical considerations to aid

- *Jian Zhao is with the University of Waterloo, Waterloo, ON N2L 3G1, Canada. E-mail: jianzhao@uwaterloo.ca.*
- *Shenyu Xu is with the Georgia Institute of Technology, Atlanta, GA 30332 USA. E-mail: shenyuxu@gatech.edu.*
- *Senthil Chandrasegaran is with the Technische Universiteit Delft, 2628 Delft, CD, The Netherlands. E-mail: r.s.k.chandrasegaran@tudelft.nl.*
- *Chris Bryan and Aditi Mishra are with the Arizona State University, Phoenix, AZ 85004 USA. E-mail: {chris.bryan, amishr45}@asu.edu.*
- *Fan Du is with Adobe Research, San Jose, CA 95110 USA. E-mail: fdu@adobe.com.*
- *Xin Qian is with the University of Maryland, College Park, MD 20742 USA. E-mail: xinq@umd.edu.*
- *Yiran Li and Kwan-Liu Ma are with the University of California, Davis, CA 95616 USA. E-mail: {ranli, klma}@ucdavis.edu.*

Manuscript received 25 Apr. 2021; revised 3 Sept. 2021; accepted 17 Sept. 2021. Date of publication 24 Sept. 2021; date of current version 30 Dec. 2022.
This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, the U.S. National Science Foundation under Grants IIS-1741536, IIS-1528203, and OAC-1934766, and an Adobe gift fund.
(Corresponding author: Jian Zhao.)
Recommended for acceptance by B. Preim.
Digital Object Identifier no. 10.1109/TVCG.2021.3114211

1077-2626 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

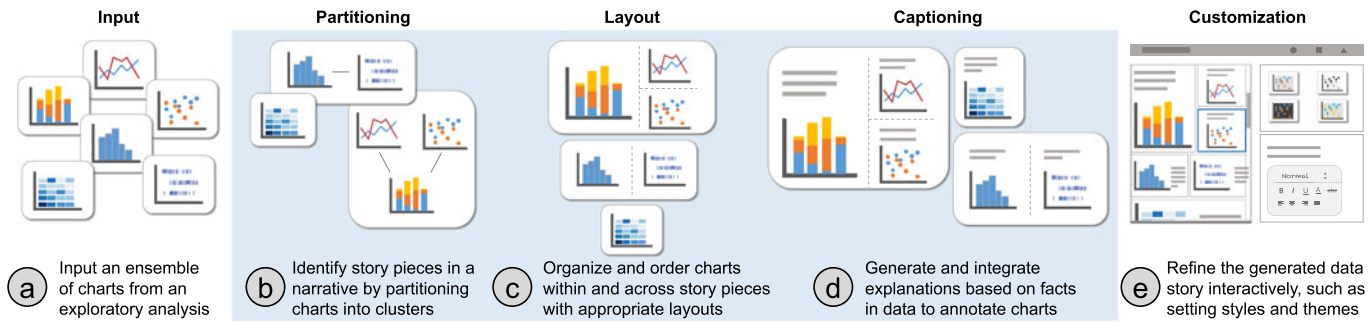


Fig. 1. ChartStory takes an ensemble of user-created charts (a) and automatically generates a data comic (b-d). This is achieved through a back-end pipeline with operations to identify story pieces (b), organize & order story pieces (c), and generate & integrate explanations (d). The user can also interactively edit the captions and layout of the charts, and change the style or appearance of the data comic (e).

the analysts in conveying their findings to their target audience.

To fill in the gap, we address the challenges of identifying and presenting a narrative through ChartStory, a tool that helps analysts automatically generate *data comics*—a major genre of visual data storytelling—from an ensemble of charts created during exploratory data analysis (Fig. 1a). As an initial step, we consider data comics, first defined by Zhao *et al.* [58, p. 2], as "*multiple visualizations (juxtaposed) into comic strip layouts consisting of a sequence of panels, each appropriately annotated and decorated with both visual and textual elements, and arranged into a sequence that progressively develops the overarching story told in the comic*," although traditional comic strips might include more embellishments. ChartStory operationalizes a set of design requirements that we term *Granularity*, *Relatedness*, *Explanation*, and *Presentation* (GREP), synthesized from prior research in visual data storytelling [22], [28]. It generates data comics by identifying "story pieces" [28] from the ensemble of charts (Fig. 1b), organizing story pieces in a layout based on data comic design patterns [8] (Fig. 1c), and generating natural-language captions to explain detected data facts (Fig. 1d).

To achieve this automatic pipeline (Figs. 1b, 1c, and 1d), we leverage advanced analysis of chart specifications (such as marks, channels, and transformations) to algorithmically group and organize an ensemble of charts into appropriate layouts. We also automatically extract data facts from the charts and use natural-language generation (NLG) techniques to annotate the data comics. ChartStory provides a set of easy-to-learn interactions that allows users to edit, customize, and refine the generated data comic layout and captions (Fig. 1e). By operationalizing established principles for identifying and presenting narrative segments from charts, ChartStory provides a good "first cut" of narration that the analyst can further edit and reorganize as needed. While several aspects of ChartStory apply to other storytelling mediums (e.g., infographics), we focus on data comics as they improve reader focus and engagement [51]. We focus on the *specific* type of data comics defined by Zhao *et al.* [58].

We evaluate ChartStory on its ability to aid comic-style narrative presentations of data analysis results from three aspects: *authoring*, *readability*, and *overall use*. The first, a controlled study, evaluates the ease of authoring, comparing ChartStory against a baseline version with no automated grouping or layout abilities. The study simulates the scenario of "handoff" in collaborative analysis [55], [57] where analysts share their analysis results with a team member who assimilates and presents the results. The second study evaluates the readability of the grouping and layouts of data comics generated in the first study. Finally, a third interview study with two data scientists shows that—when using their own charts created from their own analyses—ChartStory is able to convey a useful and presentable narrative.

In summary, our contributions in this paper include:

1) A set of requirements (abbreviated as GREP) derived from prior work [22], [25], [28] to operationalize data-driven storytelling,
2) An automated pipeline that operationalizes GREP, as well as an interactive system named ChartStory , for partitioning, layout, and captioning an ensemble of charts to generate a data comic, and
3) An evaluation that illustrates the value of our approach to both data comic creators and consumers.

The code, associated data, and supplementary materials can be accessed at https://github.com/WatVis/ChartStory.

2 Background

2.1 Challenges in Visual Presentation of Data

The importance of a seamless transition from analysis to presentation was emphasized almost right at the inception of visual analytics. Thomas & Cook recommended that effective tools require *production*, *presentation*, and *dissemination* [46]. Many of the existing tools focus mainly on production, rather than presentation and dissemination. For example, computational notebooks (e.g., Jupyter [26]) are versatile media allowing the reproduction of analytic workflows. However, sharing and presenting these analyses still requires exporting the results and displaying visualizations using a different format.

Some early research focuses on the automated generation of charts to present relational information, exemplified by Mackinlay's APT [29]. This focus still exists, as evidenced in recent and more sophisticated tools. For instance, Voyager [52] provides a ranked recommendation of automatically-generated charts based on selected data variables. CompassQL [53] is a query language that groups similar visualizations and selects representatives from each group. Draco [31] is a formal model that represents visualizations as facts and design guidelines as constraints. These tools focus on

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

choosing the right visualization by encoding design knowledge and practices for creating visualizations.

Our main contributions lie in presentation and dissemination. ChartStory helps analysts communicate their findings as visual data-driven stories, using the comic-style medium. We draw from theoretical and empirical studies in this area [8], [22], [28], and operationalize the extracted principles into an automatic pipeline that recommends partitioning, layout, and captioning, when creating data comics.

2.2 Data-Driven Storytelling and Data Comics

Analysis of a curated collection of recent stories [44] presents a set of design intents: communicating a narrative and explaining data, linking separated story elements, enhancing structure and navigation, and providing controlled exploration. Several works focused on theoretical conceptualizations of these intents. For instance, Generalized Space-Time Cubes [5] displays both existing and unrealized temporal narrative designs as a projection, flattening, or unfolding of a space-time hypercube. Brehmer *et al.* [10] proposed a design space for timeline-based storytelling using as dimensions the layout, scale, and representation of the narrative. Based on this design space, Timeline Storyteller [11] was designed as an authoring tool for presenting event sequence data. Ellipsis [37], another authoring tool, helps create interactive narrative visualizations as a combination of scenes, annotations, and user interactions. Our approach is similar in this aspect: the system automatically develops "story pieces" [28] based on the input set of charts or visualizations (i.e., partitioning).

Data comics specifically have been identified as a form that—through the combination of familiarity, established conventions, and potential for expressive freedom—can serve as an effective medium to engage the reader, convey complexity, and enable decision making [7], [50]. Bach *et al.* [6] used graph-based storytelling exercises to identify design factors for creating what they call "graph comics". The factors concern different aspects of graphs, their mapping to a comic-style storytelling paradigm, and their comprehension by the average viewer.

DataComicsJS [58] is a Chrome plugin that allows users to clip existing visualizations and compose, render, and narrate them in the style of a comic-book narrator. More recently, Datatoon [24] offers a unified pen-and-touch-based interactive authoring environment for analysis and presentation of networks. Datatoon's narrative design component provides a canvas for storyboarding and composing data comics. It also suggests basic comic layouts based on temporal/spatial continuity and filters. Our work is close to Datatoon in the application area, with the following main differences: (1) ChartStory is an automated data comic generation system, while theirs is an authoring tool focusing more on expressivity, and (2) ChartStory can form narratives from any set of data charts provided they are derived from the same dataset, while theirs only focuses on narration of graph-based visualizations.

2.3 Data Story Generation and Presentation

There has been a movement in the digital humanities to provide an orthogonal classification system that categorizes different artists and the layouts of their works [9]. On the other

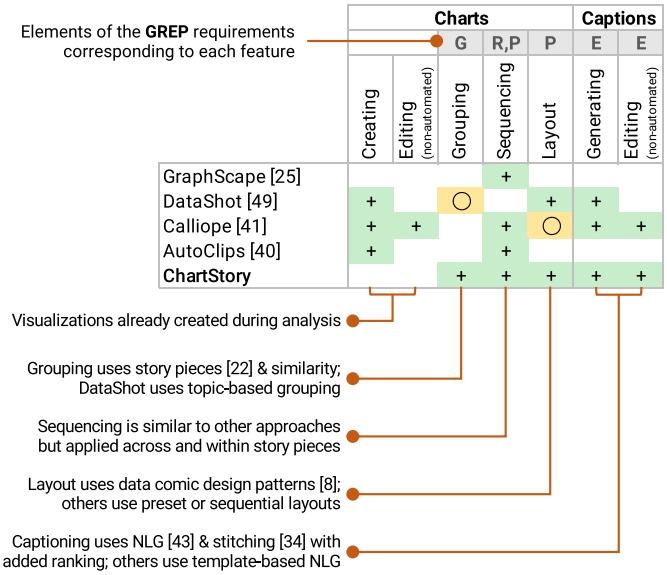


Fig. 2. Feature comparison between data story creation systems. A + symbol indicates that the feature is present, while a ◯ symbol indicates a partial addressing of the feature. Automation is implied for all features except for editing of charts and captions. The callouts below the table explain the differences between the corresponding features in ChartStory compared to the other systems.

end, algorithmic and graphical approaches have focused on *generative* aspects, such as identifying keyframes from a continuous film to select and lay out a comic-book "adaptation" [15] or simply generating, scaling, and placing text annotations within a comic [14]. These approaches have not been limited to comics. Computational support in the form of suggestions and refinements has also been pursued in graphic design. For instance, DesignScape [32] uses an energy-based model, sample styles, and design constraints (e.g., symmetry, alignment, and overlap) to suggest candidate layouts given a set of graphical elements on a page.

Within narrative visualization, the interest has tended towards methodological and algorithmic approaches for sequence detection, layout design, and caption (or annotation) generation. A general comparison between ChartStory and existing key systems for generating narrative visualizations is provided in Fig. 2. GraphScape [25] suggests a sequence of narration by computing a cost based on chart transitions, data filtering operations, and people's preference for consistent chart subsequences. ChartStory differs from GraphScape in several aspects (Fig. 2): (1) ChartStory leverages Hullman *et al.*'s theory [22] and chart transition cost to cluster charts and generate story pieces, which is not considered in GraphScape; (2) ChartStory further lays out the story pieces based on data comics design patterns [8], instead of a linear sequence. Zhao *et al.* [58] used elements of comics such as rendering style, characters, and captions to propose ways to generate narratives from existing visualizations. Srinivasan *et al.* [43] suggest data facts to which the user can refer for deciding on an appropriate visualization, but do not focus on the layout in storytelling.

DataShot [49] is a system that automates "fact sheet" generation by grouping data facts together into topics, and mapping the topics to graphical representations learned from a set of sample infographics. The narrative is thus

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.



Fig. 3. The *GREP* design requirements illustrated in its application to a data story generation system from a set of user-created charts.

driven by the data facts and embellished by the charts. ChartStory on the other hand generates visual story pieces by grouping charts generated by analysts, and—based on the chart specifications—creates layouts and narratives that are driven by the charts and embellished by data facts (Fig. 2). QuickInsights [17] provides a formulation of "interesting patterns" given a dataset and implements a technique to mine them. Calliope [41] extends the approach in DataShot using a Monte Carlo tree search algorithm to explore story pieces and present them in a logical order given tabular data as input, then using this order to animate transitions in AutoClips [40]. However, they used basic predefined layout templates based on a simple sequential layout. While a tiled layout is possible, it simply involves "wrapping" the sequential panels to fit on a page, and does not take advantage of the comic panel medium. In contrast to Calliope and QuickInsights, ChartStory uses charts—created by an analyst during data analysis—as inputs, identifies story pieces, and lays them out according to data comic design patterns [8] (Fig. 2). Chen *et al.* [13] proposed a workflow for converting analysis results into a visual storytelling layout. Our work differs from theirs in the following aspects: (1) ChartStory automatically generates annotations by using a novel language-stitching technique [34] including coreference, subordination, and conjunction patterns, while annotations are generated based on templates in their case, (2) ChartStory integrates design principles for data storytelling into an automated layout generation, while theirs focuses on a basic timeline or force-directed layout.

3 Design Rationale

For ChartStory, we base our approach on theoretical principles that have been formalized [8], [22], [28], [44] for the generation, sequencing, and layout of narrative visualizations. Based on these principles, we identify a set of requirements that we term *GREP* (Fig. 3)—Granularity, *R*elatedness, *E*xplanation, and *P*resentation—for designing an automated system that generates data comics [58] from a set of user-created charts to present a visual data story [28]. In articulating these requirements, we use the term *"panel"* in the sense of a panel in a comic book, but more specifically as a combination of a *single* chart with annotations and/or text explanation(s), contained as one unit.

Granularity: Stories can be separated into *microstructures* [47]. In narrative visualization, specific facts supported by data are what form the microstructure. These facts or sets of facts are called "story pieces" [28] and would form the "sequence of panels" in Zhao *et al.*'s [58] definition of data comics. The system should identify individual story pieces based on data variables, markup, and/or visual representations extracted from a collection of charts resulted from data analysis.

Relatedness: In contrast to microstructures, links relating events to each other form *macrostructures* [47]. To create a narrative flow, connections or "content relations" [8] need to be established *within* and *between* story pieces. These relations would help the "progressive development" of the story as defined by Zhao *et al.* [58]. Hullman *et al.* [22] characterize connections as "transitions"—differences between visualizations that help in creating a linear narrative. These are categorized into *implicit transitions*, inferrable from the data attributes/variables, and *explicit transitions* that depend on the author's interpretation. While explicit transitions lie beyond the scope of this work, implicit transitions—such as a set of related visualizations sharing similar variables—can be automatically inferred. The system should characterize implicit transitions within and between story pieces to present a meaningful narrative.

Explanation: In the comic form, connections between panels are implicitly shown through a combination of "repetition, variation, and contrast" [19, p. 12]. Other connections in the panels are explicitly shown using text, in the form of dialogues or descriptions. For data comics, the narrative can be conveyed *implicitly* by showing changes along one data dimension while preserving overlap between panels—the "progressive" component in Zhao *et al.*'s definition [58]—or *explicitly* by generating text explanations that supplement the visualizations [22]—the "textual elements" in the definition. The system should provide sufficient explanation for the narrative both implicitly and explicitly.

Presentation: Panel layout is as important to creating a narrative in data comics as content relation [8]. For instance, a linear panel layout may be best for conveying temporal changes in a narrative, while a tiled layout may convey complementary information more effectively. Appropriate selection of panel layout and sequencing can reduce visual complexity, structure information, and aid understanding [51]—the "visual element" in Zhao *et al.*'s definition [58]. The system should thus also choose a panel layout and sequence for the story pieces that best convey the narrative. Since automated narrative generation systems are not perfect, the system should keep the human in the loop, giving them the freedom to reorder items within the micro and macrostructure, alter the presentation style, and even tweak the data comic generation parameters.

These requirements stem from the idea of a story being built as a series of narrative tasks, such as (1) identify and group charts that are similar across multiple attributes except one, to form granular (*G*) transitions [22], (2) show relations (*R*) between charts by comparing and contrasting [8], (3) generate text explanations (*E*) based on differences between grouped charts, and (4) use existing patterns in data comic designs [8] to present (*P*) the grouped charts.

4 Usage Scenario

Based on GREP, we design and develop ChartStory for automating the process of crafting a data comic from a set

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.



Fig. 4. ChartStory interface showing (a) Data comic panel, an advanced panel with two tabs: (b) Structure overview and (c) Parameter setting, and a configuration panel with three tabs: (d) Page setting, (e) Themes, and (f) Content editing. High-resolution images can be found in our online supplementary materials: https://github.com/WatVis/ChartStory.

of charts created during the exploratory data analysis. In this section, we use a simple scenario to demonstrate the usage of the tool with a real-world dataset.

Suppose Jack, a data analyst working for an online retail store, needs to explore a dataset of the store's online visitor records and present his findings to a group of key stakeholders. Jack conducts an exploratory visual analysis using an analytics tool on the dataset and creates several insightful charts. However, he is worried about presenting his findings in a compelling, cogent, and elegant narrative. Composing such a narrative needs considerations of charts' layout, order, appearance, as well as many other aspects, but the analytics software does not offer this function and Jack has little background in graphics design. While there exist some authoring tools for creating data stories, unfortunately, he needs to present his results by the end of the workday and does not have the time to learn these tools.

Thus, Jack imports his charts into ChartStory . Within seconds, a comic-strip style data story is automatically generated for him (Fig. 4a). He observes that ChartStory identifies three coherent topics based on his findings in the charts, which are organized into three tiers (i.e., a collection of comic panels) [12, p. 50].

He sees that the tiers and layouts mostly make sense. The first tier mainly shows observations around customer profiles. The main chart describes the distribution of customer visits on different `cloth sizes`, and three smaller charts display three other perspectives about the distribution of customer visits, i.e., visits by `preferred color`, `date`, and `country`. Both the second and the third tiers describe insights around customers' product purchases. The two charts of the second tier show the distribution of the purchases at two different aspects, `referrer channel` (e.g., search_engine) and `page type` (e.g., cart), whereas the two charts of the third tier indicate more details about those purchases by `referrer channel` in terms of two dimensions, `hour` and `page type`. However, for the first tier, Jack wants to present the last chart of the three equal-sized smaller charts first, because it indicates the fact that compared to other countries, the U.S. has the most customer visits, which is the first point he wants to present to the stakeholders about customer files. Jack can easily switch these two charts in ChartStory .

Jack also notices that ChartStory automatically generates captions for each chart and seamlessly integrates them into the panel layout. The text is mostly based on interesting facts observed from the underlying data of the charts, along with some contextual information for the terms. He browses the text and uses a side panel to further add or remove the facts for a few charts via a list providing all the available facts (Fig. 4f). Then, ChartStory automatically updates the corresponding captions, a natural language paragraph describing the chosen facts. After completing the adjustment of layouts and contents, he focuses on refining the presentation styles, such as aspect ratio and font (Figs. 4d and 4e).

With the compelling data story, Jack gives the following presentation: *"Looking at the first tier, the female customers are more active over the male customers and the US has the most customer visits. In detail, our customers have preferences on small-sized clothes and clothes with bright colors such as yellow and some dark colors such as black, blue, and grey. The number of our customer visits has an outburst around two weeks before Christmas. In terms of customers' purchase counts, we can see that most purchases are referred from two major referrer channels. Our customers tend to make purchases from the product pages, the home page, and the cart page. There is also an unexpected temporal pattern that customers from one of the major channels tend to make purchases during the evening while those from the other channel commonly checkout very late at night."*

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

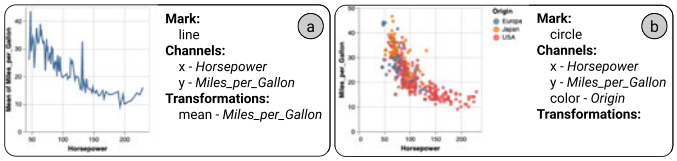


Fig. 5. Characterizing charts: properties that are common and different between two charts are used to measure their distance and identify implicit story transitions.

5 ChartStory System

The above scenario is made possible in ChartStory through a series of operations (Fig. 1). That includes *partitioning*: identification of story pieces from the overall narrative, *layout*: arrangement of charts within story pieces and ordering of story pieces themselves, *captioning*: generation of explanations based data facts, and *refining*: stylization of the overall visual theme and adjustment of other aspects. Please refer to our supplementary materials for more information.

5.1 Partitioning: Identify Story Pieces

As previously mentioned, stories exhibit levels of granularity and are composed of *microstructures* (G). From a collection of charts created by a user, story pieces—subsets of charts semantically coherent and similar to each other (e.g., charts expressing similar facts and ideas)—are identified in ChartStory as the first step.

To characterize the coherence between the charts, we define a metric that measures the distance between a pair of charts based on prior work [25], [31], [39], [55]. Specifically, motivated by VegaLite [39] and GraphScape [25], we represent individual charts as a set of specifications: *Marks* (e.g., bar, point), *Channels* (x-position, y-position, color, size), and *Transformations* (e.g., sort, aggregate). For example, Fig. 5 shows two charts characterized with these three aspects. As the input to ChartStory , we assume that the specifications are meta-data information associated with the charts. If charts are not created with VegaLite, methods on reverse engineering charts into the VegaLite can be applied [33].

The distance between two charts is quantified by calculating the sum cost for a set of operations to "transition" between the two charts' specifications, based on our prior work [55]. Considered operations include *Add*, *Modify*, and *Remove*. In Fig. 5, to transition from the left chart to the right chart, the mark type is *modified* from line to circle, a color channel is *added*, and a mean aggregate transformation is *removed*. Each operation is assigned a numerical cost value based on the results of Kim *et al.*'s [25] empirical study. They also introduced a directed graph model of the chart design space, GraphScape, in which nodes represent chart specifications and edges represent the operations with weight denoting the cost. Thus, the total transition cost (i.e., distance) between two charts is computed by summing the edge weights along the shortest path traversed from one chart to another. Unfortunately, GraphScape for real-time calculation is quite expensive (and non-interactive) due to running a breath-first search for finding the shortest path in a large graph (i.e., the chart design space). To achieve an interactive distance calculation, we add together the individual operation costs in light of our previous work [55], which takes linear time.

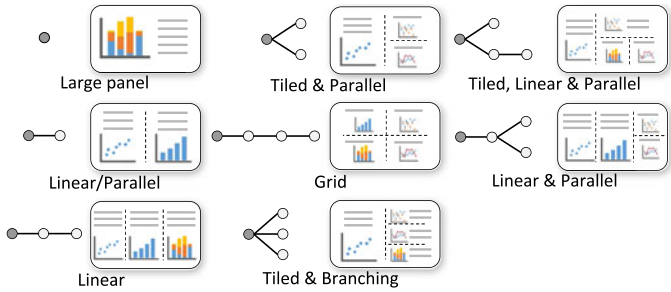


Fig. 6. Matching story backbones to Bach *et al.*'s layout patterns [8].

Based on the calculated pairwise distances, hierarchical agglomerative clustering (HAC) [20] is used to separate the charts into groups or clusters, which we consider as the identified story pieces. In the ChartStory prototype system, we limit the maximum cluster size to four per traditional comic book design patterns [30], where the four-panel comic format is quite popular [4]. We also ground this approach in patterns identified by Bach *et al.* [8], whereby individual story pieces tend to contain few elements for the sake of readability and simplicity. ChartStory uses the average-linkage metric to measure cluster distance, but other metrics, thresholds, and clustering techniques can easily be adopted as future work.

5.2 Layout: Organize and Order Story Pieces

While the previous step groups the charts into distinct story pieces, the charts within each piece do not *a priori* contain any inherent ordering or organization. To present the charts with a comic-style narrative, ChartStory automatically organizes them in a two-dimensional layout (Fig. 6) according to the connection and relation of their contents (R).

Within each story piece, we construct a weighted graph of the charts with edges indicating the distances, inspired by Hullman *et al.*'s [22] graph-driven approach for understanding visual narratives. Since every pair of charts in the graph has a calculated distance, these are complete graphs. For each graph, we compute a minimum spanning tree (MST) to find the subsets of edges that connect all the charts with the smallest total cost. We set the root node of each MST as the simplest chart (i.e., with the fewest specifications), based on the assumption that these charts often represent the most high-level, coarse-grained, and/or initial explorations. In this case, a walk-through from the root node to a leaf node likely follows the Visual Information-Seeking Mantra [42], starting from an overview of data and driving down to details. If multiple charts have the same specification complexity, we select the earliest-created chart, assuming that users, in general, explore the data progressively. We also fine-tune the MST by prioritizing charts with the same set of attributes on the paths from the root to leaf nodes.

Our layout method is grounded by the approach of maintaining consistency in visualization sequences with an objective function (minimizing the state transition cost) applied on a weighted graph of charts [22]. This MST—which we call the *story backbone*—represents a narrative structure such that one can present a story piece with the highest possible continuity by following the edges. The edges themselves represent *implicit transitions* (R) between charts. Based on the distance measure, charts with similar

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

encodings and shared attributes will be more likely connected in the story backbone, thus easing the transition in storytelling. This addresses the *implicit narrative* requirement (*E*).

There are eight different combinations of story backbones for a story piece with four charts or fewer. We map story backbones to Bach *et al.*'s [8] design patterns for data comic layout in the presentation space (*P*). Each story piece is laid out using one of these design patterns as a tier. One story backbone might be reasonably matched with multiple layouts. Currently, however, there are no agreed rules or principles for automated matching. Thus, based on Wang *et al.*'s [49] survey for layouts in infographics, we adopt frequently-used layouts such as *Tiled* (40.4%) and *Parallel* (28.6%), while omitting rarely-used ones such as *Network* (2.0%) and *Annotated* (2.9%). We also consider the space efficiency of the layouts, e.g., avoiding too small charts and increasing the data-ink ratio [48]. Fig. 6 shows our mappings between story backbones and layouts.

Similarly, we build a weighted graph between story pieces using the root charts from each story piece, and then find the shortest path that connects them. We linearly order story pieces based on this shortest path, forming the complete story that reflects its *macrostructure* (*G*).

5.3 Captioning: Generate and Integrate Explanations

As seen in infographics and comic strips, a compelling story requires not only an *implicit narrative* in the form of layout and sequencing but also an *explicit narrative* in the form of text explanations (*E*). ChartStory automatically generates text that describes insights for a chart, directly from the chart's data. Based on Srinivasan *et al.*'s work [43], we can extract a large set of data facts for each chart. For example, a data fact "The Car with a horsepower of 193 bhp has the lowest `Miles_per_Gallon` (9)" can be extracted from Fig. 5a. The data fact is characterized with meta-properties including *fact form*: `minimum`, *fact level*: `1`, and *data attributes*: `Horsepower`, `Miles_per_Gallon`, that are defined in [43]. The challenge then is to select an appropriate subset of data facts to support the explicit narrative.

Our first improvement is to propose a ranking algorithm of data facts based on the coordination between charts. For a chart with a list of data facts $C^s = [f_0^s, f_1^s, \ldots, f_n^s]$, the weight of a data fact $f_i^s \in C^s$ is defined as:

$$w_i^s = \sum_{C^t \in N(C^s)} \frac{1}{|C^t|} \sum_{f_k^t \in C^t} \alpha |F_k^t - F_i^s| + \beta |L_k^t - L_i^s| + \gamma J(A_k^t, A_i^s), \tag{1}$$

Here, $N(\cdot)$ represents the neighboring charts in the story backbone; F, L, and A are the data fact meta-properties: form (e.g., correlation), level (i.e., level 1, 2, or 3), and data attributes of a data fact; $J(\cdot)$ is the Jaccard distance; and α, β, and γ are parameters that we set equally in our implementation. Intuitively, this algorithm measures how much a data fact for a chart is related to all the data facts in the neighboring charts within the story piece. If a data fact for a chart tends to have the same type, level, and data attributes as those for other charts, it will be ranked higher. Based on the initial ranked lists of data facts, we discount duplicated facts that appear in later charts in a story piece.

TABLE 1
Examples of Language Stitching

Stitching Pattern (1): Coreference
Bulleted: • Explosives/ Bombs/ Dynamite (weaptype1) has the highest Attack Count (938) for 2015 (iyear). • Explosives/ Bombs/ Dynamite (weaptype1) has the second highest Attack Count (840) for 2014 (iyear).
Stitched: The weapon type of explosives/ bombs/ dynamite has the highest attack count of 938 for the year of 2015 and also has the second highest attack count of 840 for the year of 2014.
Stitching Pattern (2): Subordination
Bullet: • The Attack Count for Private Citizens & Property is 328.67 times of that for Tourists; • Private Citizens & Property (targtype1) has the highest Attack Count (989).
Stitched: The attack count for private citizens & property, which is 989 as the highest, is 328.67 times that for tourists.
Stitching Pattern (3): Conjunction
Bullet: • 2004 (iyear) has the lowest Attack Count (319); • 2014 (iyear) has the highest Attack Count (3925).
Stitched: The year of 2014 has the highest attack count of 3925, in contrast, 2004 has the lowest of that as 319.

Our second improvement is to concatenate the top four data facts for each chart into a single text explanation that reads like a natural narrative. To do this, we employ three language stitching patterns commonly used in natural language generation [34]: (1) *co-reference*, which eliminates subjects that are repeatedly mentioned in multiple data facts, (2) *subordination* (e.g., "which", "that"), which links data facts that are dependent of each other, and (3) *conjunction* (e.g., "while", "however"), which establishes a correlative or contrast relationships between data facts. This concatenated explanation is then integrated into the layout generated in the previous step for presentation (*P*). Fig. 4① and ③ show a comparison between the original data facts and stitched explanations. Table 1 shows more examples. We also retrieve relevant information on specific terms from Wikipedia to provide context and aid understanding, accessible via a hyperlink on the term (Fig. 4②).

5.4 Refining: Edit and Style Stories

Our goal is not to completely replace human effort with automation, but to reduce the effort toward visual storytelling by pruning the design space and recommending aspects of data story creation. While ChartStory automatically generates a data comic from a collection of input charts, it also supports a set of lightweight interactions for fine-tuning the presentation of the final data comic (*P*).

These interactions include several standard presentation and styling options, which are accessible via a Configuration Panel with three tabs (Figs. 4d, 4e, and 4f). It can be used to control page settings such as aspect ratio (Fig. 4d), chart styles such as light/dark visual appearance or themes like those of Excel and ggplot2 (Fig. 4e), and text content such as font settings (Fig. 4f).

The user can change the structure and content of the data comic if not completely satisfied with the generated results. Charts can be swapped within and between story pieces. When this happens, the text explanations (i.e., concatenated data facts) of charts are updated automatically. At the macrostructure level, story pieces themselves can be reordered as well. The user can choose data facts other than the

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

displayed ones by choosing from a ranked list shown in the second tab on the Configuration Panel (Fig. 4d). The data fact text can also be directly edited in place.

Finally, ChartStory provides an Advanced Panel with two tabs (Figs. 4b, 4c) that show the story backbones as trees, allowing for a better understanding of the data comic generation process. The first tab (Fig. 4b) allows a user to add or remove charts for generating the data comic (i.e., directly and indirectly editing the story backbones). The second tab (Fig. 4c) allows an expert user to tune the parameters (e.g., weights and thresholds) of the generation process for specific goals (e.g., increasing β in Eq. (1) to have more level-consistent explanations). To reduce the complexity, we expect this panel to be hidden for most use cases.

6 Evaluation

To assess ChartStory, we characterize a set of four research questions that correspond to ChartStory's partitioning, layout, captioning, and styling operations.

- *Q1.* *Partitioning:* Given an ensemble of charts: (a) does ChartStory automatically cluster charts in a way that helps the creator identify story pieces, and (b) does each story piece convey a narrative meaning to the reader?
- *Q2.* *Layout:* Does ChartStory effectively organize charts within story pieces (and the story pieces themselves) into a comic-style layout such that: (a) the creator needs little or no reorganization to convey the intended narrative, and (b) the reader can make sense of the narrative flow within and between story pieces?
- *Q3.* *Captioning:* Within each story piece, do the generated and integrated explanations for each chart: (a) help the creator in providing explanations to each panel and/or story piece, and (b) help the reader understand the narrative and context of each story piece?
- *Q4.* *Styling:* Does ChartStory allow the user to customize the appearance of the data comic such that (a) the creator can convey their intended theme through the styling, and (b) the theme is understood by the reader?

Using *Q1–Q4*, we conduct a trio of evaluations. *Study #1* is a controlled user study that investigates the *creator*'s perspective on the *authoring* aspect, by assessing the required effort and the resultant satisfaction when creating data comics. In contrast, *Study #2* investigates the *consumer*'s perspective on the *readability* aspect, by assessing if ChartStory's automated pipeline provides results comparable to what can be done manually. Results from these two studies show several strengths of ChartStory 's automated pipeline, though the *captioning* module (which in Study #1 originally generated data facts only in a list format without stitching) was notably criticized. This led to the second improvement on Srinivasan *et al.*'s work [43] described in Section 5.3 of translating discrete data facts into a natural paragraph based on the stitching patterns. Finally, *Study #3* provides an assessment of the *overall use* of ChartStory based on a pair of interview studies with data scientists who create and review data comics using their own datasets (see the online supplementary materials for additional information).

To provide a reasonable comparison of ChartStory 's automated operations in Studies #1 and #2, we developed a manual version of ChartStory called *Baseline*. Baseline provides access to the same set of charts and data facts as ChartStory but omits the automated backend pipeline (Fig. 1b–1d). To create a data comic, the user manually performs all authoring steps: placing charts into story pieces, organizing charts into a layout within each story piece, and ordering story pieces. Provided data facts are ordered by level only (default approach in [43]), as opposed to the ranking (by the story piece structure) and stitching procedure outlined in Section 5.3. While graphics editors such as Adobe Illustrator could serve as a baseline, they have a steep learning curve and have significantly different interfaces.

6.1 Study #1: Evaluation With Data Comic Creators

Study #1 is designed for data comic creators, to measure the user experience of authoring data comics with ChartStory . As mentioned earlier, this study simulates "handoff" in collaborative analysis [55], [57], where the person presenting the results is not necessarily the person(s) who conducted the analysis. We formalized the study as a data comic-creation task for participants, using either ChartStory or Baseline . Measured study outcomes are *task performance*, *user satisfaction*, and *qualitative feedback*.

Datasets. We curated two study datasets based on an expert analysis of two real-world datasets: one regarding US colleges [1] and another regarding gun violence [3]. These datasets were chosen due to their popularity on Kaggle, a data science platform that hosts over 45,000 datasets. For each dataset, we invited a data analyst to conduct a free-form visual exploration using Jupyter Notebook [26] and generate charts to document interesting findings. The data analyst is a researcher who has three years of experience in both data analysis and visualization. She conducted an exploratory analysis of the datasets for her research. We obtained two ensembles of charts (eleven charts for the Gun dataset, and ten charts for the College dataset), which are rendered using Vega-Lite [39]. To ensure that visual complexity was not too high, charts were restricted to visualizing a maximum of three data attributes.

Participants and Apparatus. Twelve participants were recruited (eight male, four female), aged 20 to 28 ($\mu = 23.3$, $\sigma = 2.22$). All were university students (two undergraduates and ten graduate students) in engineering and science programs, with data analytics skills as an essential part of their training. Using a seven-point Likert scale, participants self-reported a high degree of comfort in performing data analysis ($\mu = 5.25$, $\sigma = 1.05$) and in reading charts and data visualizations ($\mu = 5.92$, $\sigma = 1.00$), and very high familiarity with comics ($\mu = 6.33$, $\sigma = 0.89$). The study was conducted on a 27-inch iMac computer (3.7GHz 6-core processor, 32GB RAM, 5120×2880 display resolution) using the Chrome web browser in full-screen mode.

Task and Design. We employed a within-subjects design with two independent variables: *interface* (ChartStory and Baseline) and *dataset* (College and Guns, see below), counterbalanced using a 2×2 Latin square design. The overarching task was to create a data comic using a given ensemble of charts and one of the interfaces. This task was broken up into a set of five discrete subtasks *S1–S5* (Table 2). Note that

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

TABLE 2
Subtask Descriptions and Relevant Research Question for Study 1

S1: Partitioning (Q1)
Baseline: Divide the charts into coherent narrative story pieces of less than four charts.
ChartStory: Briefly describe the narrative of each story piece.
S2: Chart Layout (Q2)
Baseline: Choose a layout from the templates for each story piece.
ChartStory: If necessary, rearrange the layout of each story piece.
S3: Captioning (Q3)
Baseline: Select less than three data facts per chart from the list.
ChartStory: Decide if the data facts serve the narrative; change them if necessary.
S4: Story Piece Ordering (Q2)
Baseline: Order story pieces to form a narrative overall.
ChartStory: If necessary, reorder story pieces to form a narrative overall.
S5: Styling (Q4)
Both: Choose a proper style to personalize the data comic.

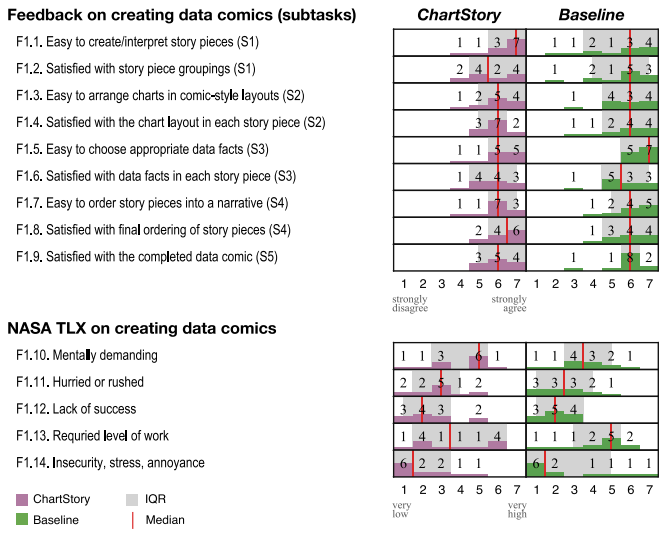


Fig. 7. Participants' responses about the subtasks (F1.1–.9, the higher the better) and the overall effort (F1.10–.14, the lower the better).

the subtask requirements differ slightly between the two interfaces: Baseline requires fully manual composition, while ChartStory initially automates subtasks *S2–S4* and allows subsequent manual refinement.

To take the study, participants created two data comics: one using ChartStory and one using the Baseline. For each interface, participants began with a training stage via a short tutorial demonstrating the interface's functionality. Then, *S1–S5* were performed on a small testing dataset (an ensemble of eight charts from the Global Terrorism dataset [2]) to acclimate participants to creating data comics using the interface. After the training stage, the chosen dataset for the interface was loaded and *S1–S5* were performed again. Next, participants filled out a short questionnaire that included NASA's TLX [21] to assess the data comic creation process with that interface. During the study, the experimenter sat beside participants to help with any confusion. After completing both interfaces, participants were invited to a short semi-structured interview for comparing the two interfaces on aspects such as ease of use and effectiveness.

6.2 Results and Analysis of Study #1

On average, the study took approximately 40 minutes for each participant to complete. As Study #1 participants did not create the charts, they needed time to familiarize themselves before beginning *S1* (reviewing time was also occasionally needed in subsequent subtasks). Because of this, *task completion time* is not analyzed as an outcome in Study #1. See Section 8 for more discussion.

To evaluate ChartStory in Study #1, we analyze three primary outcomes: task performance, user satisfaction, and qualitative feedback. At a high level, they indicate ChartStory effectively supports creating data comics in a semi-automated manner, as compared to Baseline's fully manual approach. Due to a data storage issue, log and recording files for two participants were corrupted in this analysis.

6.2.1 Task Performance

To analyze the task performance of ChartStory , we consider data points that can act as proxies for *Q1–Q3*. As users specify styling manually both in ChartStory and Baseline , task performance is not analyzed for *Q4*.

To investigate *Q1*, we compare how similar chart groups are between ChartStory and Baseline . In ChartStory, charts could not be moved between story pieces, but Baseline allowed users to place charts freely within any story piece. A high similarity between the two interfaces would indicate that ChartStory partitions charts within story pieces similar to what a user would do when allowed unrestricted placement. We compute grouping similarity using Normalized Mutual Information (NMI) [27], a well-known metric for comparing clustering algorithms with ground truth labels. However, no ground truth is available in our case, as there is no absolute correct partitioning. We thus use the clusters of Baseline as the "ground truth" to assess how much those of ChartStory differ. The average NMI of ChartStory compared to Baseline is 0.61 ($\sigma = 0.14$) in a [0,1] range where 1 represents the same clustering. This indicates that ChartStory generated chart groups relatively similar to the manual approach using Baseline .

For *Q2*, we review the number of layout edits (i.e., changing the chart layout) made to the story pieces generated by ChartStory. On average, participants made 1.2 ($\sigma = 0.71$) layout edits per story piece, with 3 and 4 story pieces in total for the Gun and College datasets, respectively. This indicates that they were generally satisfied with the layouts generated by ChartStory (Fig. 7, *F1.4*).

To answer *Q3*, we review the number of data fact edits made to ChartStory's automatically-generated data facts. On average, users made 8.8 ($\sigma = 6.5$) edits out of more than 40 data facts provided by the system. The edits were participants altering the default chosen data facts (i.e., the top 4) to display on a chart, where replacing one data fact was counted for two edits: one for removal and one for addition. They did not make any text edits to the data facts. This likewise reflects that participants were generally satisfied with the suggested data facts (Fig. 7, *F1.6*)

While each of these data points does not in a vacuum demonstrate ChartStory 's viability, they provide initial evidence that the system generates effective automated results

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

at partitioning (*Q1*), chart layout (*Q2*), and generation of data facts (*Q3*). Further investigation of user satisfaction and qualitative feedback corroborates these findings.

6.2.2 User Satisfaction

User satisfaction of the data comic creation process is analyzed using the post-study questionnaire. Fig. 7 summarizes the responses. We follow Dragicevic *et al.*'s principles [18] to report the results in histograms rather than using p-values for accuracy and transparency. In the context of assessing *Q1–Q3*, these results indicate that ChartStory 's automated features provide a level of user satisfaction similar to manual data comic creation while potentially saving time and effort. This process includes: identifying story pieces from a collection of charts (*Q1: F1.1, .2*), generating the layouts that make narrative sense in different granularity (*Q2: F1.3, .4, .7, .8*), and providing text explanations with consistent narrative and context (*Q3: F1.5, .6*). Analyzing satisfaction with the overall process (*Q1–Q4: F1.10–.14*) revealed similar distributions between ChartStory and Baseline, with participants finding ChartStory's process to be more mentally demanding, but slightly less frustrating.

6.2.3 Qualitative Feedback

Qualitative feedback from participants was collected during post-study interviews. Based on this feedback, a set of four broad sentiments was shared by participants about Study #1's data comic creation process.

ChartStory is Easier to Understand and Use. Seven participants explicitly stated ChartStory was the easier interface to understand and use (*P1, 3, 5, 7, 9, 13, 14*), while only four participants stated Baseline was easier (*P4, 6, 7, 12*). The primary justification for ChartStory 's ease is that it automates tasks and layouts. As *P1* noted, "*most things are automated. I just need to review it and update*." This was echoed by *P5*: "*ChartStory is easy to understand because it is already laid out for you. You can understand the story from each story piece.*"

Baseline Provides More Freedom. Participants who preferred Baseline generally reported its lack of constraints as the reason. *P4* stated: "*I have the freedom to make any stories that I wanted. [Baseline] was much more flexible.*" This was echoed by *P8*: "*you can group [charts] however you want*." (*P12*) liked the blank slate initially provided by Baseline : "*For the Baseline , I got a fresh page to start with... it was fresh and you could create anything as you thought*." Even for some participants who thought ChartStory was easier, they felt that Baseline 's flexibility ultimately led to more satisfying results (*P1, 7, 14*). For example, *P1* commented: "*It provided me the better results because I actually have to do everything..*"

Manually Grouping and Arranging Charts in Baseline Quickly Becomes Tedious. On the other hand, "*having to do everything manually*" (*P3*)—particularly the grouping and arranging charts—quickly became a tedious endeavor for some. *P5* summarized it thus: "*[for Baseline ,] I need to create from scratch in which case I need to spend a lot of time studying the visualizations. I had a hard time figuring out which visualization should go to which story piece.*" Similarly, *P2* disliked that Baseline "*doesn't have guidance for us to build a story.*"

Bulleted Data Facts do not Flow. Regardless of the interface, several participants mentioned the bulleted presentation of data facts hindered the narrative flow of the captions. "*The bulleted list gave me the information I wanted, but it did not have a flow*" (*P7*). Similarly, "*making it more [...] like in a word document would improve readability*" (*P5*). While participants could edit the data facts into a paragraph by deleting the list "bullets" and changing the text to flow, this was a tedious operation to manually perform for each chart, and likely contributed to the mental demand (*F1.10*).

6.3 Study #2: Evaluation With Data Comic Consumers

To complement Study #1's focus on data comic creation, Study #2 focuses on the perspective of data comic *consumers*, using the data comics created in Study #1.

Dataset. Based on the output from Study #1, we had a collection of ten data comics manually created using Baseline (five each for the College and Gun datasets). We included the "default" data comics for each dataset automatically generated by ChartStory (i.e., without subsequent user refinement), resulting in twelve total data comics. We exclude subsequent user refinement as our intent is to compare the results of ChartStory's automatically-generated comics directly against Baseline's manually-generated ones.

Participants. We recruited six expert visualization practitioners (*E1–6*) with an average of 5 years experience ($\sigma = 3.45$) in data-based storytelling. Two were data scientists (*E1, 2*) at a large digital marketing company, whose daily jobs involve reading analytics dashboards and presenting insights to help marketers design their campaigns. Four were university researchers (*E3–6*) in an Information Science department, who research domain datasets and publish academic papers to report on novel data patterns revealed in visualizations. Using expert visualization practitioners as participants allows us to adequately assess the quality of automatically- and manually-generated data comics.

Task and Design. Participants performed rating tasks for each dataset (chart ensemble). We first introduced the dataset. Then, for each of the six data comics in the dataset, participants completed a questionnaire to rate its quality in terms of partitioning, layout, captioning, and styling (*Q1–4*). Participants also provided subjective feedback about the comic's overall design. After all data comics were individually rated, participants ranked the set based on their overall preference. This process was then repeated for the second dataset. Both the order of datasets and the order of data comics within each dataset were counterbalanced to mitigate learning effects. The study lasted about 1 hour.

6.4 Results and Analysis of Study #2

We report two types of results for Study #2: questionnaire ratings (again following Dragicevic *et al.*'s principles [18]) and qualitative comments given about the data comics. To provide an initial assessment, we compared the overall rankings of ChartStory and Baseline charts (Fig 8, *F2.6*). On average, participants gave a higher (better) ranking for ChartStory's automated charts ($M = 2.5$, $IQR = [1.5, 3]$) than the manual Baseline charts ($M = 4$, $IQR = [2, 5]$).

6.4.1 Q1: Partitioning

Overall, participants preferred the partitioning of ChartStory ($M = 5.5$, $IQR = [4.5, 6]$) compared to Baseline ($M = 5$,

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

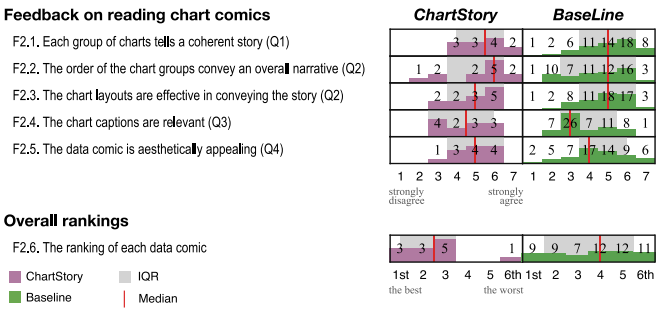


Fig. 8. Participants' feedback about reading chart comics (F2.1–.5, the higher the better) and the overall ranking (F2.6, the lower the better).

$IQR = [4, 6]$) (Fig. 8, *F2.1*). The most common issue with Baseline lies at its choice of variables. For example, when reading a story about the Guns dataset, *E3* asked: *"Why did it use month as x-axis here but then switched to age in the next chart?"* Similarly, *E1* commented: *"The variables are too diverse and not focused on one story... I cannot see common patterns."E1* also pointed out an issue that several story pieces in user-created comics had only one chart, and said *"this does not tell a story at all."*

Five out of six participants thought ChartStory did a better job with choosing a cohesive set of variables. *E5*: *"Most of the groups have a theme."* *E1*: *"Good clustering... the axes are related to each other and consistent."* *E3*: *"I could see consistent x and y between charts in each story piece."E2* did not express a preference toward ChartStory or Baseline but complained that *"he could often find a chart not sharing any variable with others"*. One repeated suggestion (*E1, 3, 6*) for improving ChartStory was to combine similar story pieces. *"The charts in story pieces 2 and 3 all have age as x-axis and should be grouped together"* (*E3*). Only *E4* suggested splitting a story piece: *"I am not sure if it's good to show external factor (location) with personal factor (age) in one story piece... this misleads me to think they are correlated"*.

6.4.2 Q2: Layout

When evaluating the order between story pieces, most participants strongly preferred the comic created using ChartStory ($M = 6$, $IQR = [4, 6]$) over Baseline ($M = 5$, $IQR = [3, 6]$) (Fig. 8, *F2.2*). One observed reason for this is that the comic created using ChartStory *"orders the stories from simple to complex, from overview to details"* (*E3*). *E3* further explained that ChartStory 's data comic for the College dataset *"first reports more macro variables like month and sex, and then reports distributions over categories that are non-binary like age and place."* Similarly, *E1* was pleased that, *"I can see transitions between story pieces, very good!"* Several participants proposed additional ordering rules to fit their workflow. *E5*: *"I prefer to always have region-related charts in the first story."* *E4*: *"key performance indicators such as admission rate should be presented first."*

ChartStory ($M = 5$, $IQR = [4, 6]$) and Baseline ($M = 5$, $IQR = [4, 6]$) received similar ratings when evaluating the layout of the charts within each story (Fig. 8, *F2.3*). Participants observed that data comics created using Baseline tended to use a linear sequential layout while ChartStory resulted in a more diverse structure. *E1* preferred ChartStory 's layout because *"the charts could be read in parallel and back-and-forth."* He pointed at a Baseline result and noted, *"it is unclear why one chart needs to be placed after or below another."* *E5* also liked the data comic created with ChartStory: *"It looks very compact and uses the space more efficiently."* However, this was not always the case. *E3* preferred Baseline because *"it looks simpler, like a conventional document."E6* complained about ChartStory: *"The order within each story piece is unclear,"* and suggested annotating the order explicitly.

6.4.3 Q3: Captioning

Overall, participants did not show a strong preference between the text explanations in ChartStory ($M = 4.5$, $IQR = [3, 5.5]$) and Baseline ($M = 3$, $IQR = [3, 5]$) (Fig. 8, F2.4). *E1* found text explanation helpful in both contexts, as *"the text can make sure that people are on the same page and getting the same insights and conclusions from the same chart."E2* added that *"when the chart shows too much data or the trends are not easy to see, captions can make me aware of them."* Similarly, *E4* mentioned that *"the text loses some information but captures the key signals with explanations."*

All participants agreed that the captions can be improved. Many found that some of the data facts were not informative and not easy to read, as *E1* explained: *"The caption says what is high, what is low... I can see it in the chart."* To mitigate this issue, *E1* and *E4* suggested emphasizing data facts that *"you could not see by eye"* were not directly evident in the charts, such as the trends and comparisons: *"Looking at the bar chart, I think the comparison between bars and the trend of growth give more insights than the counts, which help me tell a better story."* As a suggestion for improvement, *E6* asked for story piece-level captions (as opposed to captions appended to individual charts) that provide *"global context about the shared key variable across the charts."*

To enhance the readability of the captions, *E1* proposed: *"Instead of showing a ranked list, we can group the facts by topics or order by overall to detail to tell a story, like the captions in news articles."* Similarly, *E2* suggested that *"the bullet points are not super readable... I want something more natural that I can read aloud to my colleague."*

6.4.4 Q4: Styling

While all the data comics shared similar visual styles (chart types, mark and channel encodings, color schemes, etc.), participants found ChartStory ($M = 5$, $IQR = [4, 6]$) more aesthetically pleasing than Baseline ($M = 4$, $IQR = [4, 5.5]$) (Fig. 8, *F2.5*). For example, when reviewing Baseline's data comics, *E2* complained that some charts *"are in different sizes and hard to compare"*. At one point, *E1* pointed at a story piece and noted: *"The charts are overlaid by mistake... better make sure the charts are aligned and axes are aligned."* In contrast, none of these defects were observed with ChartStory. *E1, 3, 6* individually applauded that it looks polished and uniform.

Suggestions were made to improve both systems regarding the choices of visual encodings and color schemes: *"same variables should be encoded consistently and different variables should be encoded differently"* (*E1*), *"I don't like the stacked bar chart of month versus number of records...the values are very close "* (*E5*), and *"the heatmap is impossible to read... hard to differentiate green, yellow, blue, and the gradients in between"* (*E6*).

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

Strategies were also proposed for better utilizing the space. For example, *E3* suggested that *"simpler charts should get less space... the heatmap should get more space because it has more information."E4* asked to allocate more space to important story pieces, and *E6* suggested that *"similar charts should get even sizes and (be) placed side by side...it makes the comparison easier."* These comments underscore how ChartStory is ideal for use by analysts who perform the analysis and want to communicate their findings. By automating the data comic generation, ChartStory not only provides a way of conveying the story but also allows the analyst to reflect on aspects of their analysis/visualization that may be absent from their collection of charts. This allows them to iterate between analysis and presentation, with very little effort spent on the presentation itself.

6.5 Study #3: Interview Studies With Domain Data

As a follow-up to Studies #1 and #2, which looked at data comic creation and consumption in discrete, controlled settings, we conducted an interview study to holistically assess ChartStory in the context of real-world data applications. For this, we coordinated with two data scientists from Study #2 (*E1–2*), who work at a large digital marketing company. The data scientists mainly use Vega and Vega-Lite as the main charting tool in their daily workflow. They provided a domain marketing dataset of an online retail store, along with a set of nine charts created using Voyager [52]. Voyager was chosen because it is one of their frequently used tools and it can easily export the charts in Vega-Lite for the input to ChartStory .

The marketing dataset includes 27,780 customer profiles (e.g., `gender`, `country`, and `cloth_size`) and their online journeys (e.g., `referrer_channel`, `product_views`, and `purchase_count`). The provided charts were made to analyze the performance of the online store, visualizing the growth of the traffic and revenue, the composition of the customers, and customers' common behavioral patterns.

ChartStory generated a data comic containing four story pieces from the charts. The first story piece consisted of four bar charts, separating customers by `cloth_size`, `preferred_color`, `gender`, and `country`. The second showed a scatterplot of `gender` by `device_type`. The third showed two side-by-side barcharts of `purchase_count` categorized by `referrer_channel` and `product_views`. The last one used two scatterplots to further break down `referrer_channel` and `product_views` by `hour_of_day`, with the sizes of the dots encoding `purchase_count`.

Upon seeing the generated data comic, *E1* and *E2* immediately remarked on the value of the automated data comic creation. *E1* contrasted the data comic to more linear presentation formats (such as data notebooks): *"It is much more meaningful to look at the stories than just a bunch of charts ordered by file names or creation time."* Interestingly, *E2* considered the charts as a form of presentation dashboard: *"The stories are very cohesive... good dashboard."* After reviewing the story pieces, both suggested combining the first two story pieces, since the charts involved different characteristics of the customer profiles. *E2* also suggested combining story piece 3 and 4 (since they share many variables), but *E1* disagreed: *"I prefer to keep them separate since they are at different levels of detail."* Both *E1* and *E2* agreed that ChartStory provides a good starting point for organizing charts, requiring only minor manual refinements. This was succinctly noted by *E1*: *"I am surprised that it is fully automatic... it will definitely save me and my colleagues a lot of time."*

Studies #1 and #2 revealed to us that the bulleted presentation of data facts was frustrating at demonstrating a coherent story narrative. Therefore, for this study, we employed the "language stitching" technique (see Section 5.3 and Fig. 4). Both participants, who had previously seen the bulleted presentation in Study #2, reported preferring the new text explanations. *E1* said: *"I like the full sentence captions... it is a better form to tell a story... bullet points are less formal to read"*. *E2* commented: *"The language is more natural... it is closer to the captions I need in my reports and slides... this feature makes my job easier."*

At the end of the interview, both participants expressed their excitement about ChartStory and asked for long-term deployment studies. They also requested additional features, including generating story-level captions, supporting filters for exploring data comics, and adding effects to make the exploration more interactive.

7 Discussion

Structured by the research questions *Q1–Q4*, we discuss the findings from our studies and their implications on semi-automated data comic creation and consumption.

Q1: Partitioning. Study #1's survey responses indicated that ChartStory 's solution was as high as Baseline's manual grouping, even though ChartStory did not allow chart switches across story pieces (which was disabled for the study to reduce confounds). The experts' feedback in Study #2 indicated this constraint was a good idea: participants' manual grouping of story pieces often went against some fundamental visualization and storytelling principles, such as maintaining continuity across plots in a story piece. ChartStory, with its distance-based approach to partitioning, avoids this pitfall, and allows for the creation of additional metrics for a more consistent and coherent output.

Q2: Layout. Using ChartStory, participants from Study #1 were able to achieve an equivalent level of satisfaction with arranging charts in a significantly shorter amount of time. They felt the lack of layout constraints in Baseline was an advantage: *"I can do what I want to layout and organize the story"* (*P2*). However, we observed that with no constraints, all participants violated design principles for comic strips, such as arranging comics in non-space-filling layouts (with significant white space) and positioning adjacent cells in uneven alignments, even though half of them were exposed to ChartStory first in the study. Fig. 9 compares the data comics by participants using ChartStory with the ones using Baseline. The layouts in Figs. 9b, 9e is close to the initial automated output (Figs. 9a, 9d), while Fig. 9c, f show haphazardly-arranged charts created using Baseline. This difference was also noticed in Study #2. ChartStory was mostly preferred because of the story ordering (simple to complex), ease of comparison between charts, compact layout, and transitions between story pieces; however, Baseline was also preferred because it followed a more conventional "document-like" layout.

Q3: Captioning. The initial caption generation technique in Studies #1 and #2 was based on Srinivasan *et al.*'s work

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.



Fig. 9. Data comic examples of the College (a, b, c) and the Gun (d, e, f) datasets. (a, d) The default data comics generated by ChartStory. (b, e) The finished data comics by participants using ChartStory. (c, f) The finished data comics by participants using Baseline. High-resolution images can be found in our online supplementary materials: https://github.com/WatVis/ChartStory.

[43], with the data facts ranked based on the relationships between charts within a story piece. In both studies, our ranking technique did not fare differently from the original method [43] that is used in Baseline. However, participants found it difficult to read captions in the form of separate bulleted items, though they acknowledged that the data facts provided a good starting point for captioning. Our language stitching approach (Section 5.3) was better received in our follow-up studies, who stated that the captions were closer to what they would use. Moving forward, we plan to explore natural-language generation approaches that can synthesize higher-level captions based on data facts prioritized by the user. This could help address the issues of low-level and high-level storytelling uncovered in the study.

Q4: Styling. Participants from Study #1 tended to find it tedious to give the data comic a finished look using Baseline, as dragging and resizing the charts to fit the available two-dimensional layout space required manual sizing and placement. As a result, data comics created using Baseline looked slightly unfinished (e.g., Fig. 9c), in contrast to the layouts automatically generated by ChartStory. This difference was also noticed by experts in Study #2, who preferred the *"polished and uniform"* look of data comics created by ChartStory, compared to those created using Baseline. In Study #2, the main criticisms of ChartStory involved functionalities also shared in Baseline, such as color palette and granularity of data encoding in the charts themselves. Control of such parameters is given to the user currently, though it is feasible to specify rules for such attributes which will further ensure uniformity and aid easy creation of data comics without sacrificing the quality and aesthetics.

Broader Implications. Besides the above aspects specific to each research question, our development and evaluation of ChartStory deals with several generalizable takeaways regarding human-machine collaboration. First, the design of automated tools should focus on assisting users with tasks that they do not excel, to improve the overall productivity. With ChartStory, analysts can focus on creating charts (that they are good at) to better serve the story and worry less about layout or presentation (that they are not as good at). This is similar to, for example, typesetting using LaTeX, which allows the writer to focus on content instead of document layout and styling.

Second, although auto-generated results may not be perfect—as evidenced by study users performing subsequent manual refinements in Studies #1 and #3—this approach can provide a serendipitous benefit for users by enlightening them while improving their efficiency. As an example, in normal practice, a user has to manually write the textual annotations for a data comic, which is a time-consuming process. ChartStory automatically generates text items, thus provides a good starting point for further refinement requiring much less effort to arrive at the "final" polished version of the data comic.

Third, there is a trade-off between user freedom and constraint in developing semi-automated tools. As discussed earlier, while our study population claimed high familiarity with comics, they showed poor skill at actually designing them. ChartStory operationalizes the design patterns by Bach *et al.* [8] to help users achieve a good design by constraining their operations. In comparison, Baseline users—in spite of having complete flexibility—created sub-optimal layouts. However, too much constraint may result in issues in trust, which is another factor to consider. Determining the appropriate balance between freedom and constraint is beyond the scope of the current work.

8 Limitations and Future Work

While the study results indicate ChartStory is promising to help analysts generate compelling data comics. The system and our study design still have limitations.

First, the MST currently prioritizes chart (attribute) similarity based on the implicit transitions proposed by Hullman *et al.* [22]. However, visually similar charts need not always show related data or be part of the same narrative. This problem could be exacerbated by scale: more charts to partition results in a higher chance of more unrelated charts clustered together. The MST we propose is illustrative and not comprehensive, and it is a robust enough technique that can adapt to new measures of similarity or "relatedness." For example, with advanced machine learning techniques, an embedding

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

space of charts could be learned using a similar method by Zhao *et al.* [56], so that the relatedness can be better captured. This flexible notion of relatedness is worth exploring in the future. Also, enlightened by the theories of Cohn [16], more studies can be conducted to investigate users' strategies of reading the data comics with different panel layouts and how such relatedness plays a role in their comprehension. These studies will further shed light on understanding the effect of comic design layout patterns in ChartStory in assisting storytelling, compared to the "random facts" in Wang *et al.* [49].

Second, ChartStory may be limited in processing a large collection of charts, which results in a large number of story pieces. However, it does not make sense for an analyst to present a large number of charts in a single narrative, which overwhelms their audience and defeats the purpose of storytelling. Some algorithms may be developed to help select the most relevant charts before generating a data comic, or rank different data comics generated from the charts based on certain criteria. Also, ChartStory limits each story piece with a maximum of four charts, based our observations of Bach *et al.*'s patterns [8] and the prevalence of four-panel comic [4]. Future studies could be conducted on investigating the effect of story piece size in ChartStory .

Third, Study #1 suffered from the limitation that participants were not the people who performed the data analysis and created the charts in the first place. While the study was indeed based on the existing scenario of handoff in asynchronous collaborative analysis [55], [57], the limitation was also imposed for the sake of uniformity—allowing users to create their own charts would have resulted in the number and relevance of the charts affecting the outcome and the narrative. By keeping the input charts uniform across participants, we mitigate any variations that may be caused by individual differences in the analysis. Also, we introduce the issue of comprehension—some participants may be able to read and interpret the input charts quicker than others. Thus, we did not use "data comic creation time" as an indicator of the usefulness of ChartStory. Study #2 involved a relatively small number of participants, an unavoidable constraint when participants are required to be experts/practitioners. We plan to conduct an additional evaluation of ChartStory to address these issues.

Lastly, we focus on a specific type of data comics defined by Zhao *et al.* [58]. While the outputs of ChartStory already resemble several attributes of general comic strips (e.g., panel layout and annotation), certain aspects are missing. For example, comic strips usually have characters to carry out the story, speech bubbles to narrate information, and special visual effects to emphasize story elements. However, past studies indicate that data comics do not usually have a character [7], [8]. To resemble the speech bubbles, the generated explanations of ChartStory can be placed in-situ on the charts when appropriate. It is also worth exploring techniques that use special effects to emphasize certain parts of a chart, such as magnifying outliers or calling out key patterns with distortion-based or picture-in-picture methods. Further, augmenting ChartStory with richer authoring functionalities, such as allowing users to add embellishments, could address this limitation and provide more flexibility, but a trade-off between automation and manual editing needs to be considered in the design.

9 Conclusion

We have presented ChartStory , a tool that helps analysts craft a data story in comic-strip style from charts generated in their visual exploration of data. ChartStory provides an analytical pipeline to automate the partitioning, layout, and captioning of data comics, as well as several intuitive interactions to allow for further refining and styling. The design of ChartStory is iteratively improved, grounded by a set of design rationale (GREP) distilled from the literature. We conducted a comprehensive evaluation of ChartStory by comparing against a manual baseline with data comic creators and consumers, followed by in-depth interviews with two data scientists. The results indicated that ChartStory can provide cogent recommendations for creating data comics that make narrative sense to participants and compare favorably to data comics created by the baseline.

Acknowledgments

Jian Zhao, Shenyu Xu, and Senthil Chandrasegaran contributed equally.

References

[1] College Scorecard Data, 2019. [Online]. Available: https://collegescorecard.ed.gov/data
[2] Global Terrorism Database | Kaggle, 2019. [Online]. Available: https://www.kaggle.com/START-UMD/gtd
[3] Gun Deaths in the US: 2012–2014 | Kaggle, 2019. [Online]. Available: https://www.kaggle.com/hakabuk/gun-deaths-in-the-us
[4] Why 4-Panel Comics Now Dominate Our Screens | Wired, 2019. [Online]. Available: https://www.wired.com/story/four-panel-webcomics/
[5] B. Bach, P. Dragicevic, D. Archambault, C. Hurter, and S. Carpendale, "A descriptive framework for temporal data visualizations based on generalized space-time cubes," *Comput. Graph. Forum*, vol. 36, pp. 36–61, 2017.
[6] B. Bach, N. Kerracher, K. W. Hall, S. Carpendale, J. Kennedy, and N. Henry Riche , "Telling stories about dynamic networks with graph comics," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2016, pp. 3670–3682.
[7] B. Bach, N. H. Riche, S. Carpendale, and H. Pfister, "The emerging genre of data comics," *IEEE Comput. Graph. Appl.*, vol. 37, no. 3, pp. 6–13, May/Jun. 2017.
[8] B. Bach, Z. Wang, M. Farinella, D. Murray-Rust , and N. H. Riche, "Design patterns for data comics," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2018, pp. 1–12.
[9] J. A. Bateman, F. O. Veloso, J. Wildfeuer, F. H. Cheung, and N. S. Guo, "An open multilevel classification scheme for the visual layout of comics and graphic novels: Motivation and design," *Digit. Scholarship Humanities*, vol. 32, no. 3, pp. 476–510, 2017.
[10] M. Brehmer, B. Lee, B. Bach, N. H. Riche, and T. Munzner, "Timelines revisited: A design space and considerations for expressive storytelling," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 9, pp. 2151–2164, Sep. 2017.
[11] M. Brehmer *et al.*, "Timeline storyteller: The design & deployment of an interactive authoring tool for expressive timeline narratives," in *Proc. Comput.+J. Symp.*, 2019, pp. 1–5.
[12] R. Chapman, *Drawing Comics Lab: 52 Exercises on Characters, Panels, Storytelling, Publishing & Professional Practices*. Beverly, MA, USA: Quarry Books, 2012.
[13] S. Chen *et al.*, "Supporting story synthesis: Bridging the gap between visual analytics and storytelling," *IEEE Trans. Vis. Comput. Graphics*, vol. 26, no. 7, pp. 2499–2516, Jul. 2020.
[14] W.-T. Chu and C.-H. Yu, "Optimized speech balloon placement for automatic comics generation," in *Proc. Int. Workshop Interactive Multimedia Mobile Portable Devices*, 2013, pp. 1–6.
[15] W.-T. Chu, C.-H. Yu, and H.-H. Wang, "Optimized comics-based storytelling for temporal image sequences," *IEEE Trans. Multimedia*, vol. 17, no. 2, pp. 201–215, Feb. 2015.

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.

[16] N. Cohn, "Navigating comics: An empirical and theoretical approach to strategies of reading comic page layouts," *Front. Psychol.*, vol. 4, 2013, Art. no. 186.
[17] R. Ding, S. Han, Y. Xu, H. Zhang, and D. Zhang, "Quickinsights: Quick and automatic discovery of insights from multi-dimensional data," in *Proc. ACM Int. Conf. Manage. Data*, 2019, pp. 317–332.
[18] P. Dragicevic, "Fair statistical communication in HCI," in *Modern Statistical Methods for HCI*. Berlin, Germany: Springer, 2016, pp. 291–330.
[19] T. Groensteen, *Comics and Narration*. Jackson, MS, USA: Univ. Press Mississippi, 2013.
[20] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*. San Mateo, CA, USA: Morgan Kaufmann, 2011.
[21] S. G. Hart and L. E. Staveland, "Development of NASA-TLX (task load index): Results of empirical and theoretical research," *Advances Psychol.*, vol. 52, pp. 139–183. 1988.
[22] J. Hullman, S. Drucker, N. H. Riche, B. Lee, D. Fisher, and E. Adar, "A deeper understanding of sequence in narrative visualization," *IEEE Trans. Vis. Comput. Graphics*, vol. 19, no. 12, pp. 2406–2415, Dec. 2013.
[23] M. B. Kery and B. A. Myers, "Exploring exploratory programming," in *Proc. IEEE Symp. Vis. Lang. Hum.-Centric Comput.*, 2017, pp. 25–29.
[24] N. W. Kim *et al.*, "DataToon: Drawing dynamic network comics with pen+touch interaction," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2019, pp. 1–12.
[25] 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.
[26] T. Kluyver *et al.*, "Jupyter notebooks-a publishing format for reproducible computational workflows," in *Proc. 20th Int. Conf. Electron. Publishing*, 2016, pp. 87–90.
[27] A. Lancichinetti, S. Fortunato, and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks," *New J. Phys.*, vol. 11, no. 3, 2009, Art. no. 033015.
[28] B. Lee, N. H. Riche, P. Isenberg, and S. Carpendale, "More than telling a story: Transforming data into visually shared stories," *IEEE Comput. Graph. Appl.*, vol. 35, no. 5, pp. 84–90, Sep./Oct. 2015.
[29] J. Mackinlay, "Automating the design of graphical presentations of relational information," *ACM Trans. Graph.*, vol. 5, no. 2, pp. 110–141, 1986.
[30] S. McCloud, *Understanding Comics: The Invisible Art*, New York, NY, USA: William Morrow, 1994.
[31] D. Moritz *et al.*, "Formalizing visualization design knowledge as constraints: Actionable and extensible models in draco," *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 1, pp. 438–448, Jan. 2019.
[32] P. O'Donovan, A. Agarwala, and A. Hertzmann, "DesignScape: Design with interactive layout suggestions," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2015, pp. 1221–1224.
[33] J. Poco and J. Heer, "Reverse-engineering visualizations: Recovering visual encodings from chart images," *Comput. Graph. Forum*, vol. 36, pp. 353–363, 2017.
[34] X. Qian, E. Koh, F. Du, S. Kim, and J. Chan, "A formative study on designing accurate and natural figure captioning systems," in *Proc. Extended Abstr. ACM Conf. Hum. Factors Comput. Syst.*, 2020, pp. 1–6.
[35] D. Ren, M. Brehmer, B. Lee, T. Hollerer, and E. K. Choe, "ChartAccent: Annotation for data-driven storytelling," in *Proc. Pacific Visualization Symp.*, 2017, pp. 230–239.
[36] A. Rule, A. Tabard, and J. D. Hollan, "Exploration and explanation in computational notebooks," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2018, pp. 1–12.
[37] A. Satyanarayan and J. Heer, "Authoring narrative visualizations with ellipsis," *Comput. Graph. Forum*, vol. 33, pp. 361–370, 2014.
[38] A. Satyanarayan and J. Heer, "Lyra: An interactive visualization design environment," *Comput. Graph. Forum*, vol. 33, no. 3, pp. 351–360, 2014.
[39] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer, "Vega-lite: A grammar of interactive graphics," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 1, pp. 341–350, Jan. 2017.
[40] D. Shi, F. Sun, X. Xu, X. Lan, D. Gotz, and N. Cao, "Autoclips: An automatic approach to video generation from data facts," *Comput. Graph. Forum*, vol. 40, pp. 495–505, 2021.
[41] D. Shi, X. Xu, F. Sun, Y. Shi, and N. Cao, "Calliope: Automatic visual data story generation from a spreadsheet," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 453–463, Feb. 2021.
[42] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *The Craft of Information Visualization*. New York, NY, USA: Elsevier, 2003, pp. 364–371.
[43] 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. Graphics*, vol. 25, no. 1, pp. 672–681, Jan. 2019.
[44] C. D. Stolper, B. Lee, N. H. Riche, and J. Stasko, "Emerging and recurring data-driven storytelling techniques: Analysis of a curated collection of recent stories," Microsoft Res., Redmond, Washington, USA, Tech. Rep. MSR-TR-2016–14, 2016.
[45] K. Subramanian, J. Maas, and J. Borchers, "Tractus: Understanding and supporting source code experimentation in hypothesis-driven data science," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2020, pp. 1–12.
[46] J. J. Thomas and K. A. Cook, "Illuminating the path: The research and development agenda for visual analytics," IEEE Comput. Soc., Los Alamitos, CA, USA, Tech. Rep. PNNL-SA-45230, 2005.
[47] P. W. Thorndyke, "Cognitive structures in comprehension and memory of narrative discourse," *Cog. Psychol.*, vol. 9, no. 1, pp. 77–110, 1977.
[48] E. Tufte, *Visual Display of Quantitative Data*. Cheshire, CT, USA: Graphics Press, 1983.
[49] Y. Wang *et al.*, "Datashot: Automatic generation of fact sheets from tabular data," *IEEE Trans. Vis. Comput. Graphics*, vol. 26, no. 1, pp. 895–905, Jan. 2020.
[50] Z. Wang, J. Ritchie, J. Zhou, F. Chevalier, and B. Bach, "Data comics for reporting controlled user studies in human-computer interaction," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 967–977, Feb. 2021.
[51] Z. Wang, S. Wang, M. Farinella, D. Murray-Rust , N. H. Riche, and B. Bach, "Comparing effectiveness and engagement of data comics and infographics," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2019, pp. 253:1–253:12.
[52] 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. Graphics*, vol. 22, no. 1, pp. 649–658, Jan. 2016.
[53] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer, "Towards a general-purpose query language for visualization recommendation," in *Proc. ACM Workshop Hum.-in-the-Loop Data Analytics*, 2016, pp. 4:1–4:6.
[54] H. Xia, N. H. Riche, F. Chevalier, B. D. Araujo, and D. Wigdor, "DataInk: Direct and creative data-oriented drawing," in *Proc. ACM Conf. Hum. Factors Comput. Syst.*, 2018, Art. no. 223.
[55] 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, pp. 75–86, 2018.
[56] J. Zhao, M. Fan, and M. Feng, "Chartseer: Interactive steering exploratory visual analysis with machine intelligence," *IEEE Trans. Vis. Comput. Graphics*, to be published, doi: 10.1109/TVCG.2020.3018724.
[57] J. Zhao, M. Glueck, P. Isenberg, F. Chevalier, and A. Khan, "Supporting handoff in asynchronous collaborative sensemaking using knowledge-transfer graphs," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 340–350, Jan. 2018.
[58] Z. Zhao, R. Marr, and N. Elmqvist, "Data comics: Sequential art for data-driven storytelling," Univ. Maryland, College Park, MD, USA, Tech. Rep. HCIL-2015–15, 2015.



Jian Zhao is currently an assistant professor with the Cheriton School of Computer Science, University of Waterloo, where he directs the WatVis (Waterloo Visualization) Group. His research interests include information visualization, human-computer interaction, and data science. His work contributes to the development of advanced interactive visualizations that promote the interplay of human, machine, and data.

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.



Shenyu Xu is currently working toward the PhD degree in computer science at the Georgia Institute of Technology, Atlanta, Georgia, where he works with professor Alex Endert. He is a member of the GaTech Visual Analytics Lab. His research lies on the intersection of human-computer interaction and visual analytics.



Senthil Chandrasegaran is currently an assistant professor with the Faculty of Industrial Design Engineering, Technische Universiteit Delft, The Netherlands, where he is one of the principal investigators with the Designing Intelligence Lab (DI_Lab). His research focuses on the integration of computer support tools to aid collaboration in early design, and the use of visualization and visual analytics techniques to understand how designers work together.



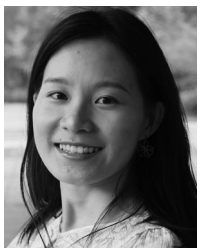
Chris Bryan received the PhD degree in computer science from the University of California, Davis, California, in 2018. He is currently an assistant professor with the School of Computing and Augmented Intelligence, Arizona State University, where he directs the Sonoran Visualization Laboratory (SVL @ ASU). His research interests include information visualization, human-computer interaction, and virtual reality.



Fan Du received the bachelor's degree from Zhejiang University, Hangzhou, China, the master's degree from the University of Maryland, College Park, Maryland, and the PhD degree in computer science from the University of Maryland, College Park, Maryland, in 2018. He is a research scientist with Adobe Research. His research interests include data visualization, recommender systems, and machine learning.



Aditi Mishra is currently working toward the PhD degree in computer science with the Arizona State University, Tempe, Arizona, where she works with Chris Bryan. She is a member of the Sonoran Visualization Laboratory. Her research focuses on visual analytics and human-computer interaction.



Xin Qian is currently working toward the PhD degree with the College of Information Studies, University of Maryland, College Park, Maryland, where she works with professor Joel Chan. She is a member of the HCIL Lab. Her research focus on natural language processing and human-computer interaction.



Yiran Li is currently working toward the PhD degree in computer science with the University of California, Davis, Davis, California, where she works with professor Kwan-Liu Ma. She is a member of the VIDI Lab. Her research focuses on visual analytics and information visualization.



Kwan-Liu Ma (Fellow, IEEE) received the PhD degree in computer science from the University of Utah, Salt Lake City, Utah, in 1993. He is a distinguished professor of computer science with the University of California, Davis. He directs VIDI Labs and UC Davis Center of Excellence for Visualization. His research interests include visualization, computer graphics, high-performance computing, and human-computer interaction. He was a recipient of the NSF PECASE Award in 2000 and the IEEE VGTC 2013 Visualization Technical Achievement Award.

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

Authorized licensed use limited to: TU Delft Library. Downloaded on January 30,2023 at 09:26:22 UTC from IEEE Xplore. Restrictions apply.