ChartSeer: Interactive Steering Exploratory Visual Analysis with Machine Intelligence

Jian Zhao, Mingming Fan, Mi Feng

Abstract—During exploratory visual analysis (EVA), analysts need to continually determine which subsequent activities to perform, such as which data variables to explore or how to present data variables visually. Due to the vast combinations of data variables and visual encodings that are possible, it is often challenging to make such decisions. Further, while performing local explorations, analysts often fail to attend to the holistic picture that is emerging from their analysis, leading them to improperly steer their EVA. These issues become even more impactful in the real world analysis scenarios where EVA occurs in multiple asynchronous sessions that could be completed by one or more analysts. To address these challenges, this work proposes ChartSeer, a system that uses machine intelligence to enable analysts to visually monitor the current state of an EVA and effectively identify future activities to perform. ChartSeer utilizes deep learning techniques to characterize analyst-created data charts to generate visual summaries and recommend appropriate charts for further exploration based on user interactions. A case study was first conducted to demonstrate the usage of ChartSeer in practice, followed by a controlled study to compare ChartSeer's performance with a baseline during EVA tasks. The results demonstrated that ChartSeer enables analysts to adequately understand current EVA status and advance their analysis by creating charts with increased coverage and visual encoding diversity.

Index Terms—Exploratory visual analysis, interactive steering, visualization recommendation, machine learning.

1 Introduction

Exploratory visual analysis (EVA) is an open-ended iterative process in which an analyst employs visual methods, such as plotting charts, to identify interesting questions and findings within data [8]. EVA is useful whenever an analyst has vague hypotheses or ill-defined tasks [32], [57]. However, EVA is often challenging to undertake because an analyst needs to continuously determine subsequent promising directions for investigation within a large parameter space [42], [65], i.e., decide which data variables to explore and which types of charts to use. Further, analysts usually fail to hold a holistic view of a current analysis, making it more difficult to make decisions to advance and steer an EVA [52]. These issues are critical not only for individual analyses but also during collaborative analysis scenarios because EVA, in practice, is often performed in multiple asynchronous sessions [67], [70].

Suppose that an analyst is exploring a dataset of product purchase records, where each record contains 15 attributes, e.g., price, volume, etc. As the analyst has some hypotheses about pricing trends, she creates a number of two-variable charts, e.g., a line chart comparing prices over time, to help her identify trends and gather insights. However, after making these charts, she soon becomes stuck and does not know which attributes to explore even though there are thousands of options that are possible (i.e., two-variable charts have 15×14 unique variable combinations with 5 potential chart marks (e.g., circle), 6 potential encoding channels for variables (e.g., position, color), and numerous potential aggregation methods (e.g., binning)). Complicating things further, she does not have any knowledge about the interrelations between charts in this large analysis space or the variables that have already been explored. This analyst's problems will become even more compounded when she hands off her half-completed analysis to her colleagues.

Researchers have attempted to address these challenges by proposing a variety of systems that automatically generate charts to serve as recommendations of future avenues for analysts to explore (e.g., [42], [58], [66]). To make recommendations useful, these systems often require some level of user guidance, such as the specification of (partial) queries or input about what attributes to explore and how to do so. Although these recommendation systems have enabled some users explore new pathways during their analysis, it is still unclear how such systems can support analysts in developing an understanding of the holistic picture being formed during their EVA so that they can explicitly (or implicitly) provide proper input to systems. Improper input currently results in unwanted, noisy recommendations that lead analysts down ineffective paths within EVA because these analysts cannot resolve uncertainties that arise [26].

To fill this gap, this work proposes *ChartSeer*, a system that helps analysts steer their EVA through the dynamic visualization and recommendation of charts. ChartSeer employs a *meta-visualization* approach [47] to present the charts that have already been created, characterizing the state of the current analysis that has already been undertaken in multiple asynchronous sessions by the same or different analysts. This approach enables an analyst to easily gain knowledge about the overall *landscape* of EVA, e.g., identifying clusters, trends, and gaps in users' behavior while creating charts or conducting an analysis. Further, an analyst could utilize this knowledge to obtain effec-

- *Jian Zhao (corresponding author) is with the University of Waterloo. E-mail: jianzhao@uwaterloo.ca.*
- *Mingming Fan is with the Rochester Institute of Technology. E-mail: mingming.fan@rit.edu.*
- *Mi Feng is with Twitter Inc. Email: fmamimi@gmail.com.*

tive recommendations by interacting with the system, e.g., peeking into "holes" in the current analysis space so that appropriate charts will be automatically generated within local regions of interest. To make this possible, ChartSeer leverages state-of-the-art deep learning models such as grammar variational autoencoders (GVAE) [37], to obtain a mapping between charts and vectors in a semantic space, and uses this mapping to create chart meta-visualizations and enable interactive recommendations. The source code associated with this work is available at https://github.com/jeffjianzhao/ChartSeer.

ChartSeer was developed based on design considerations distilled from the literature (e.g., [65], [66], [67]). To evaluate ChartSeer, an interview with a data scientist was performed to derive a use case to demonstrate the system's usefulness in advancing EVA under asynchronous collaborative settings. A controlled user study was also performed to compare ChartSeer with a basic baseline. The results demonstrated that while using ChartSeer, participants created more charts that covered more distinct data variables and encoding channels and had more diverse visual perspectives but focused more on the exploration of data. Participants' usage patterns and challenges with the system are also discussed.

2 Related Work

2.1 Visualization Similarity

Characterizing the similarity of user-authored visualizations is essential when generating an informative summary of EVA results, in both individual and collaborative analyses. Visualization similarity can provide important insights such as analysis coverage and possible directions of exploration. For example, based on a similarity metric, dimensionality reduction techniques such as MDS [36] and t-SNE [41] have be applied to produce a summary of charts in 2D space [67].

One method to measure similarity is to model visualizations using features. Image Graphs [40], for example, used parameters such as color, opacity, and so on within volume renderings to denote changes. The P-Set Model [32] extended this idea by including interactions such as zooming. Sedlmair et al. [54] proposed a concept framework by analyzing the parameter space of visualizations in the literature. Another method to measure similarity uses computes the operation or transition costs between visualizations. Hullman et al. [29], for example, proposed an objective function that minimized the transitions needed to display a sequence of charts. GraphScape [34], which used Vega-Lite [53], measured the changes between charts with respect to data transformations and visual encodings.

The above methods are largely based on the features used in characterizing visualizations or their transitions. In practice, however, it is often difficult to select the right set of features given the complexity and diversity of options possible within data charts. Researchers have explored more generic and data-driven approaches to compute similarity using machine learning techniques that derive visualization *embeddings*, i.e., vector representations in non-linear continuous spaces [18]. ChartSeer was inspired by this idea and employs deep learning techniques to map charts to semantic vectors to measure chart similarity and generate charts.

2.2 Meta-Visualization

Meta-visualization or meta-analysis [69] visually presents an ensemble of visualizations (e.g., charts). This approach has been widely used to support collaborative data analysis [15], [30]. As the present work was particularly interested in summarizing and steering in EVA, inspiration was drawn from the principle of information scent [48], which has been used in systems that are designed for both synchronous (e.g., [31], [44], [44], [56], [69]) and asynchronous analyses (e.g., [24], [59], [61], [62], [70]). Researchers have also investigated different visual cues that can be used to identify seen or unseen information so as to facilitate users' coordination and inspire future explorations. For example, treemaps and chord diagrams have been used to present the data dimensions that were explored in an analysis history [51], [52]. Dynamic graphs have also been employed to highlight the overall status of EVA by linking different components created by users such as tags, comments, and visualizations [69], [70]. Ordination, which is also used within ChartSeer, has also been used to summarize user-generated charts in a 2D space based on similarity [67].

While these systems can offer effective summarizations of EVA findings, the user still needs to manually consume and explore the synthesized information to identify future directions to explore and create new charts. This imposes cognitive and physical effort that is in addition to the effort exerted during the EVA, which can become challenging when an analysis space is vast. ChartSeer facilitates this task using machine intelligence that interactively recommends data charts to analysts for further analysis.

2.3 Visualization Recommendation

Automatically recommending appropriate charts to analysts to improve EVA, especially when the analysis space is large, i.e., consisting of numerous combinations in data variables as well as visual encodings is difficult. APT [42], for example, introduced a compositional algebra to enumerate the space of charts and rank them using expressiveness and effectiveness criteria. Gilson et al. [22] employed a mapping between data domain ontologies and visual representation ontologies to specify charts, which was later extended by VISO [60]. SeeDB [58] recommended charts by measuring their derivation from a query. CompassQL [64], the basis of Voyager [65], offered flexible query specifications for searching through a visualization space and recommended charts using heuristic rules. Along this line, Draco [46] leveraged answer set programming (ASP) to describe constraints over visualization design.

In addition, other researchers have leveraged users' analysis goals and tasks to determine effective recommendation visualizations. BOZ [11], for example, modeled low-level perceptual tasks to generate corresponding charts. SAGE [50], which extended APT [42], considered a user's goals in EVA. VizAssist [9] enabled users to interactively give feedback for a visualization automation process. User interactions have also been utilized to recommend charts. Tableau's "Show Me" [43] provided alternative visual encodings based on the selection of data attributes. Voyager2 [66] blended manual and automated chart specifications by introducing wildcards and related views to Voy-

ager [65]. Gotz et al. [23] analyzed repeated patterns in user interaction histories to infer visual analysis tasks when suggesting visualizations.

These approaches, however, were limited by the empirical tasks, perceptual models, heuristics, or rules chosen. With advances of machine learning, Data2Vis [19] attempted to build an end-to-end neural network to generate charts directly from data. DeepEye [39] combined rule-based methods with models to classify and rank visualizations. VizML [27] learned design choices from a corpus of data-visualization pairs. Compared to many of these methods, data-driven learning-based approaches allow for more generic solutions to avoid problems such as costly rule creation and the combinatorial explosion of results [27]. Therefore, ChartSeer adopted this approach and employed deep learning to convert charts to and from semantic vectors that are used for summarization and recommendation. Differing from these fully automated methods, ChartSeer supports the interactive steering of chart recommendations by integrating an analyst's creativity and domain knowledge.

3 The Design of ChartSeer

3.1 Design Considerations

The general goal of ChartSeer is to effectively steer EVA by combining humans and automation. The design of ChartSeer were informed by the principles and practices from exploratory data analysis [10], [32], [57], meta-visualization techniques [51], [67], [70], visualization recommendation [64], [65], [66], and mixed-initiative systems [26].

D1: Provide a semantic overview of charts to encourage a holistic understanding. As a basis, the system needs to provide a summary of the user-generated charts to reveal the current scope of the analysis [67], [70]. For example, arranging similar charts together provides valuable insights into the results and processes of EVA, such as trends, patterns, and outlying analysis behavior [32], [67]. As browsing multiple charts requires a higher cognitive load, the system should offer a semantic organization of the charts to enable an analyst to understand the summary in context [66]. However, charts are often complicated and contain many facets that are difficult to quantify. ChartSeer addresses this by employing deep learning to obtain a vector representation of charts in a semantic space.

D2: Suggest future avenues for exploration using cognitive and algorithmic guidance. In addition to supporting the review for already-created charts, it is critical for a system to inspire an analyst to discover unexplored and underexplored areas for further analysis. As the analysis space can be vast, the system needs to offer cognitive guidance to facilitate the discovery of additional areas to explore, e.g., through a visual summary to reveal "holes" in the current state of the analysis [51], [67]. Moreover, the system should utilize algorithmic support, e.g., recommendations of the charts appropriate for specific data subsets [64], [66]. ChartSeer utilizes both approaches to inform an analyst of possible future avenues of exploration.

D3: Allow for interactive steering to drive recommendations. As EVA is dynamic, analysts' decisions often involve assessing their current situation [57], which can be informed by the summary in D1. Thus, the system needs to support more adaptable browsing behaviors, such as interactively driving chart recommendations [65], [66]. This human-in-the-loop approach is critical to combine an analyst's domain knowledge with the strength of algorithms [26]. Inspired by these principles, a set of intuitive user interactions was implemented within ChartSeer to facilitate the dynamic steering process.

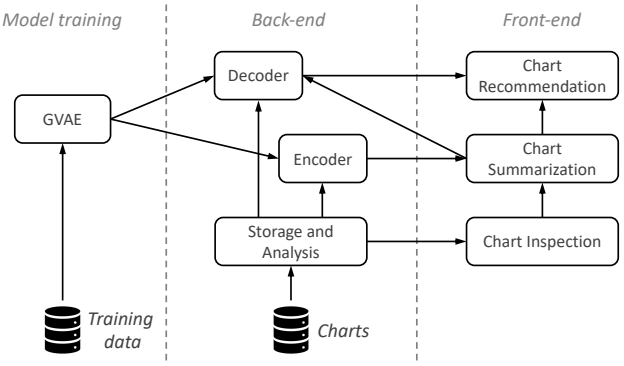


Fig. 1. ChartSeer consists of a back-end that includes a pre-trained encoder and decoder and a front-end that has three coordinated views.

D4: Support the visual investigation, manipulation, and creation of charts. To enable easiness and flexibility in EVA, the system should support the viewing and editing of user-generated charts at a lower-level, as well as system-recommended charts [65], [66]. To advance the analysis, the system should also to enable for the easy creation of new charts, either based on recommendations or from scratch [67]. ChartSeer uses Vega-Lite [53] to represent, manipulate, and create charts and offers an easy-to-learn user interface to access different functionality.

3.2 System Overview

ChartSeer's development was guided by the above design considerations and consists of two major components, (i) a back-end analytical engine and (ii) a front-end visual interface, both of which are based on a pre-trained machine learning model (Figure 1).

The back-end contains a Data Storage and Analysis module, a Chart Encoder, and a Chart Decoder. The Data Storage and Analysis module records the generated charts and their relevant information, while also handling basic computation tasks such as dimensionality reduction. The chart Encoder converts a data chart represented in Vega-Lite [53] into a numerical vector, whereas the Chart Decoder converts numerical vectors into Vega-Lite [53] data charts. The Chart Encoder and the Chart Decoder were trained using the GVAE [37] deep learning model.

The front-end is composed of three interactively-coordinated views (Figure 5). The Inspection Panel allows analysts to view, edit, and create a selected chart based on Vega-Lite. The Summarization Panel presents all the charts, including user-created and recommended charts, and the dataset. Lastly, the Recommendation Panel shows the previews of system-suggested charts.

The input charts are processed in the Data Storage and Analysis module and then fed into the Chart Encoder to generate their vector representations. These representations are used to create a meta-visualization of the charts in a 2D semantic space (**D1**; Figure 1). From this, an analyst

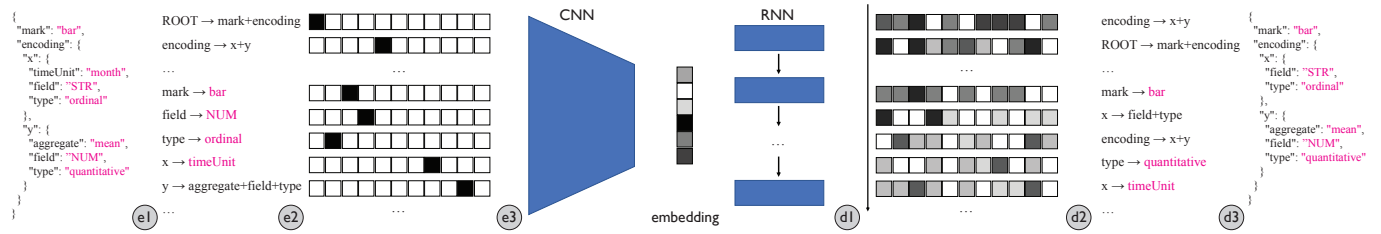


Fig. 2. ChartSeer characterizes data charts using encoding (i.e., e1, e2, and e3) and decoding (i.e., d1, d2, d3) processes that are based on GVAE.

can gain an overview of the current analysis space, identify promising directions, and leverage chart recommendations from the system (**D2**). The charts are recommended via user interactions with the system, that is, clicking on an area of interest in the 2D space (**D3**). Based on this input and the existing charts, the Chart Decoder will automatically generate a list of recommended charts. Moreover, all the user-created and recommended charts can be further inspected and tuned to create new charts (**D4**).

ChartSeer was iteratively refined over the duration of its implementation and evaluation. Based on participants' feedback and observations, the Summarization Panel was enhanced by revising the chart glyph design and adding background annotations. The backend chart analysis was also modified to generate a more stable chart summarization and the recommendation process was revised to obtain charts that were higher quality.

4 Chart Characterization

Characterizing charts with an appropriate model or representation forms the basis of informative chart summarization and recommendation (**D1**, **D2**). Inspired by the recent advances in machine learning, a data-driven *encoder-decoder* approach that converts charts to and from semantic vectors (i.e., embeddings) was employed. Based on these vectors, chart similarity was derived using the Euclidean distance and was used to compute relations, clusters, and distributions of user-created charts in the large and complex analysis space during EVA.

This approach was chosen because the embedding of features has been shown to be effective for many tasks (e.g., language translation [45]). In addition, it does not typically require an abundance of human labor to label the data or require perceptual experiments to be conducted to derive similarity. It is also more flexible and general than heuristic or rule-based methods [19] because once the embedding features are learned, they can be applied within a variety of applications. Lastly, in addition to an encoder that converts a chart into a vector, a decoder can be obtained and used to generate recommendation charts (**D2**, **D3**).

4.1 Grammar Variational Autoencoder (GVAE)

At a high-level, the notion of GVAE [37] was applied to the Vega-Lite chart representation [53]. The variational autoencoder (VAE) [35] aims to learn a bidirectional mapping between data points x in the problem space and variables z in a continuous latent space. The mapping $E(x) \rightarrow z$ is called an encoder and the mapping $D(z) \rightarrow x$ is called a decoder. In the present case, x is a chart and z is the vector representation of the chart. As shown in Figure 2,

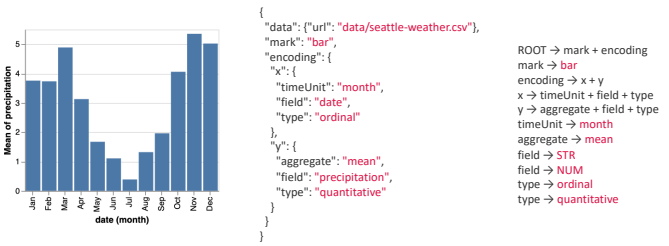


Fig. 3. (Left) A data chart. (Middle) The chart's Vega-Lite specification. (Right) The associated CFG rules.

based on VAE, GVAE aims to learn the encoder and decoder where x are parse trees governed by a context-free grammar (CFG) [33]. One benefit of a CFG-based GVAE, compared to a traditional VAE, is that it is more likely to generate valid data samples with the decoder.

As the Vega-Lite specification of charts is hierarchical, each chart can be viewed as a parse tree that is produced by a set of rules within a CFG. For example, Figure 3 shows a chart, its Vega-Lite, and the rules that generate the JSON tree. Because the goal is to employ GVAE to obtain a chart encoder and decoder independent of specific datasets, data fields in Vega-Lite are substituted with common tokens, i.e., replacing quantitative fields with `NUM` and categorical/ordinal fields with `STR`, similar to the method used in [19]. Also, specifications that are not relevant to visual encodings, such as `data` and `schema`, are excluded.

For pre-processing, a GVAE was trained to learn the chart encoder and decoder, by employing an architecture as suggested in [37]. The encoding process (Figure 2) extracts the rules forming the Vega-Lite tree **(e1)**, represents the rules with one-hot vectors **(e2)**, and feeds these vectors to a convolutional neural network (CNN) and a dense layer to map them to a latent variable (i.e., embeddings) **(e3)**. The decoding process converts latent variables back to vectors (logits) with a recurrent neural network (RNN; **(d1)**), derives the rules represented by these vectors **(d2)**, and reconstructs a Vega-Lite tree based on these rules **(d3)** .

Note that an RNN encoder is often paired with an RNN decoder [55], which is appropriate for this application as the order of **(d2)** output CFG rules affects the tree construction. However, experiments in [37] found improved performance with a CNN encoder and an RNN decoder, which may have been be due to "repetitive, translationally invariant substrings" in **(e1)** input CFG rules.

4.2 Model Training and Evaluation

During development, an alternative method was considered to learn the chart encoder and decoder, wherein each Vega-Lite specification was viewed as a text string. Data2Vis [19] employed this method to train a neural network to generate

TABLE 1
Comparison of GVAE and LSTM autoencoder.

Model	Latent Dimension	Accuracy
LSTM autoencoder	20	0.3884
	2	0.3731
GVAE	20	0.6525
	2	0.4446

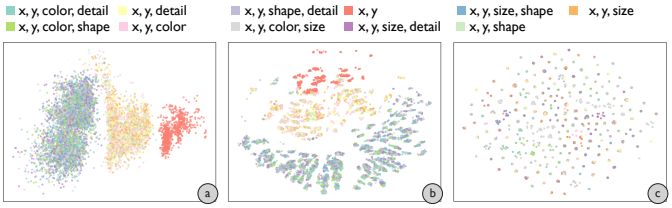


Fig. 4. A visualization of the chart embedding spaces in 2D using (1) a GVAE trained with 2-dimensional latent space, 2) an MDS projection of distances computed by GVAE trained with a 20-dimensional latent space, and 3) an MDS projection of feature-based distances [34], [67].

charts directly from data. Inspired by this idea, an RNN autoencoder based on LSTMs (long short-term memory) [25], was implemented, which learned the bidirectional mapping between the character sequences of Vega-Lite and the latent vectors. Experiments to evaluate these two models were conducted and resulted in the selection of GVAE due to its superior performance.

Based on a dataset of 4,300 Vega-Lite specifications with one to three data variables [19], charts with four variables were added using an approach similar to [49]. The final dataset contained 9,925 examples and 80% were used for training, 10% for validation, and 10% for testing.

For GVAE, an encoder containing 3-layer CNNs with a filter size of 4 and the kernel size of 3 was used, followed by a dense layer with 256 neurons. The decoder contained 3-layer RNNs based on GRUs (gated recurrent unit) [13]. Both the encoder and decoder contained 256 hidden dimensions. For the LSTM autoencoder, a similar architecture to that used in [19] was employed, with a 2-layer RNN with 512 LSTM units for the encoder and decoder. Table 1 shows the reconstruction accuracy for the two models with different latent space dimensions. GVAE performed better and had an improvement of over 25% when learning a 20-dimensional latent space. Note that this is a hard task to employ machine learning for, as data charts have many diverse aspects. GVAE, although not perfect, can provide relatively good results for analysts to start with. These promising results motivated the development of the front-end user interface.

To better understand the learned embedding space of charts, all the charts in the dataset were plotted based on 2D latent vectors produced by GVAE and color-coded using the combination of encoding channels used in the charts (Figure 4a). The plot demonstrated that even with a learned 2-dimensional latent space that has lower accuracy (Table 1), GVAE embeddings formed three distinct clusters with different encoding channels. We further compared MDS projection results from distances computed by GVAE trained with a 20-dimensional latent space and a classic feature-based method [34], [67] (Figure 4bc). To make the feature-based method dataset-independent, the same preprocessing of Vega-Lite specifications was applied as that in GVAE. This results in the MDS failing to generate notable chart clusters with the feature-based distances, whereas GVAE-based distances appeared more effective. This suggests that GVAE-based embeddings may be able to characterize the charts better in higher dimensional spaces.

5 The ChartSeer System

5.1 Chart Summarization

Based on the embedding features generated by the Chart Encoder, dimensionality reduction is used to project the charts into a 2D space from a high-dimensional feature space and then visualize them as circular glyphs (**D1**).

5.1.1 Projection and Layout

MDS [36] was employed for projection, but other dimensionality reduction techniques (e.g., t-SNE [41]) can also be applied, or an encoder with a latent dimension of two can be trained directly. The embedding only captures the visual encodings of the charts, because dataset-specific information is removed from Vega-Lite to train a generic model. However, the data information within the charts is also important for an effective meta-visualization [67] because analysts expect to see charts grouped together with similar visual representations and data variables. Thus, a weighted distance D_w is utilized (with a parameter α):

$$D_w(m,n) = \alpha|v_m - v_n| + (1-\alpha)\left(1 - \frac{f_m \cap f_n}{f_m \cup f_n}\right), \quad (1)$$

which includes the Euclidean distance of the charts' embedding vectors v_m and v_n, and the Jaccard distance between the two sets of data variables f_m and f_n.

The above distance metric reflects a trade-off between promoting *data variation* and promoting *design variation*, which is controlled by α. In two extreme cases, when $\alpha = 0$, the summarization shows clusters of charts with similar data variables, encouraging analysts to create charts with new variables to fill in the holes; when $\alpha = 1$, charts with similar visual encodings are grouped, which indicates empty spaces for new chart designs. ChartSeer sets the default value to 0.5, but allows analysts to adjust α using a slider based on their dynamic EVA goals. Figure 6 compares the projection results with different α values.

Further, a simple 2D projection can cause problems when a new chart is added or removed (i.e., during an analyst's continuous exploration of the data) because the positions of existing charts may change abruptly due to the re-computation of the projection. Fujivara et al.'s [21] method, which is based on the Procrustes transformation that finds the maximum overlap between two sets of locations, was used to minimize layout changes.

5.1.2 Visual Representation

The Summarization View in ChartSeer shows each chart as a circular glyph (Figure 5b). The inner circle displays the mark type (e.g., `L` for a line chart) of the chart and the outer circle is a donut chart with each colored segment indicating a data variable in the chart. The author ID is also attached as a badge in the top right corner. For example, as shown on the tooltip in Figure 5f, the hovered-over `Point` chart was created by `u1` with three data variables: `average faculty salary` (red), `expenditure` (green), and `control` (gray).

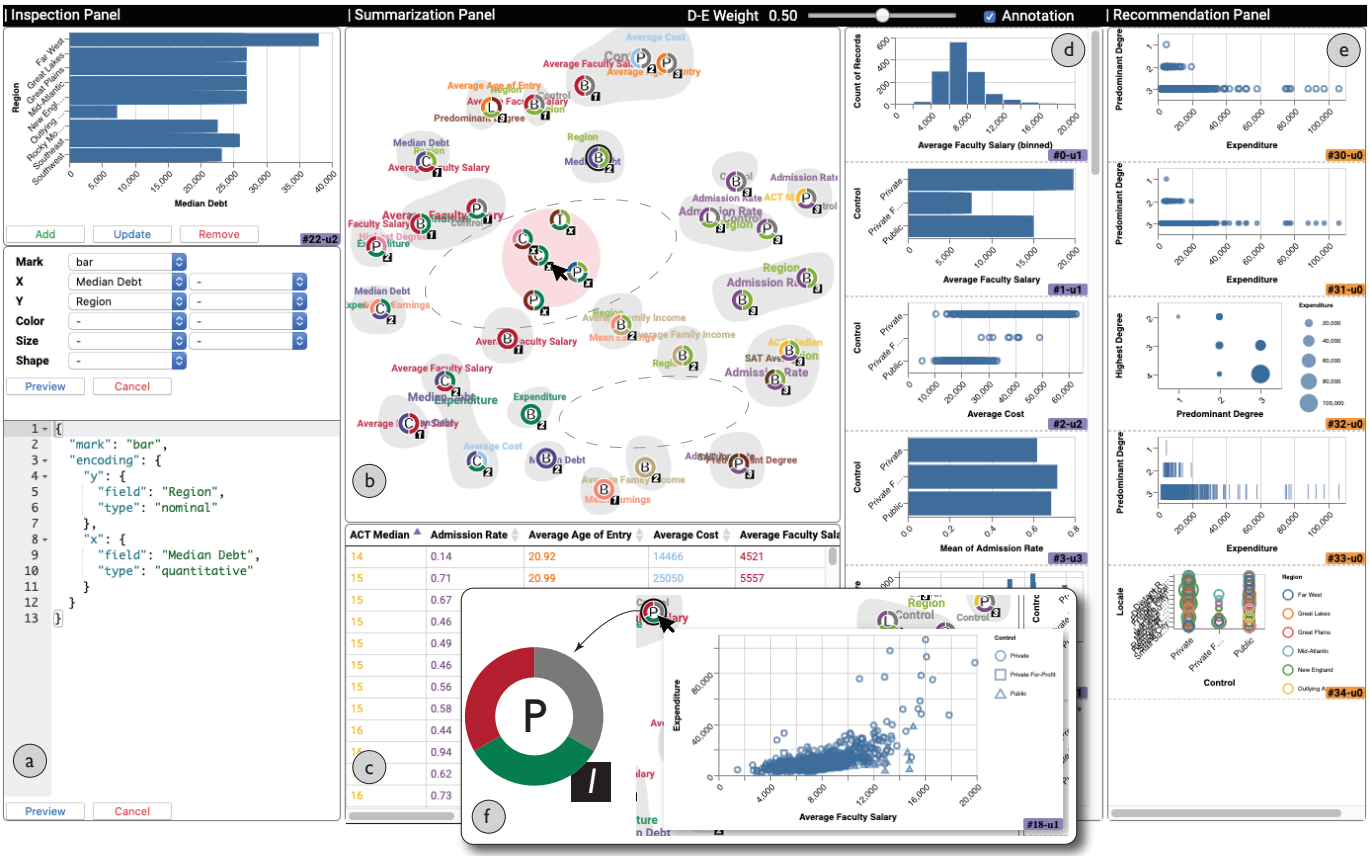


Fig. 5. The ChartSeer user interface consists of the (a) Chart Inspection Panel, Chart Summarization Panel (which includes the (b) Summarization View, (c) Data Table View, and (d) Chart List View), and (e) Chart Recommendation Panel. (f) A user is hovering over a chart.

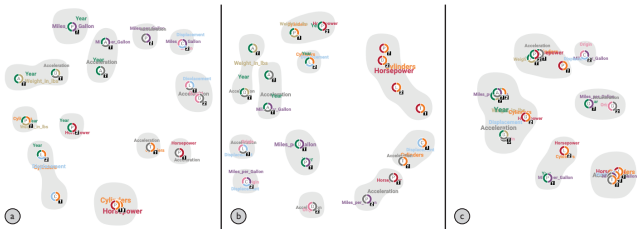


Fig. 6. Projection of charts with: (a) $\alpha = 0$ (i.e., only the distance of data variables), (b) $\alpha = 0.5$ (i.e., a weighted distance of data variables and visual encodings), (c) $\alpha = 1$ (i.e., only the distance of visual encodings).

The above glyph design aims to capture two important aspects of a chart, i.e., the visual encoding and the data variables. During the development of ChartSeer, different design alternatives were explored including a simple colored dot showing the mark type and the author ID, a pie chart showing the data variables, and a similar glyph with the inner circle showing the author ID. The current design was ultimately chosen because it enables an analyst to quickly glance at the necessary information in the chart and easily see the analysis patterns of the EVA landscape without drilling down into each chart. To further support this task, BubbleSets [14] are used to show clusters of charts in their high-dimensional feature space, similar to meta-visualization in prior work [67], [68]. Text annotations denoting the associated variables were also added, using the same color scheme of the glyph. The chart clustering is performed by hierarchical agglomerative clustering [63]. The size of the text indicates the frequency of that variable used by the charts in the cluster. These background annotations can be turned off based on a user's needs (Figure 8a).

In addition, ChartSeer offers a Data Table View of variable columns with the same color scheme (Figure 5c) and a visual list of charts to allow for quick browsing (Figure 5d), mimicking traditional interfaces such as Excel.

5.2 Chart Recommendation

From the Chart Summarization Panel, an analyst can obtain an overview of the EVA "landscape" thus far and use it to make decisions about future investigations by observing patterns from the view (**D2**). To facilitate this task, ChartSeer also supports chart recommendations using interactive steering (**D3**). Specifically, an analyst can "peek" anywhere in the Summarization View by clicking, and a set of charts will then be recommended in context, within an area indicated by a red spotlight circle that follows the cursor (Figure 5b). The recommended charts are shown with the same glyph as the existing charts, and a preview of the charts is available on the Recommendation Panel (Figure 5e). An analyst can also zoom into or out of the view or continue to click on other sections to obtain different sets of recommendations that are specified using user-interested local neighborhoods within the vast analysis space.

5.2.1 *Recommendation Process*

The chart recommendation process has the following key steps (Figure 7):

(a) Based on the user-clicked 2D position, a set of n samples $\{s\}$ is randomly selected in the projected 2D space within a radius of r (i.e., the red circle in Figures 5 and 7).

(b) For each sample s, the projection is reversed by converting the 2D position s^l back to a high-dimensional vector s^h that is understandable by the chart decoder. This is solved as the following optimization problem:

$$\arg\min_{s^h} \sum_{i=1\ldots m} \left(D_l(s,p_i) - |s^h - p_i^h|\right)^2, \quad (2)$$

$$D_l(s,p) = \frac{|s^l - p^l| - \alpha J(s,p)}{1-\alpha}, J(s,p) = 1 - \frac{f_s \cap f_p}{f_s \cup f_p}. \quad (3)$$

Similar to s^l and s^h, p^l and p^h denote the projected 2D point of a chart and its corresponding high dimensional vector. Intuitively, the objective function Eq. (2) measures the discrepancy of the high- and low-dimensional distances between the sample and all other m existing charts, where s^h is the only unknown variable. However, the low-dimensional distance $|s^l - p^l|$ in Eq. (2) cannot be used directly because in the MDS projection to the 2D space, the Jaccard distance of chart data variables in Eq. (1) is used. As the high-dimensional space and the distance $|s^h - p^h|$, which the decoder understands, contains no data variable information, the Jaccard distance term in Eq. (1) needs to be removed as indicated by D_l in Eq. (3).

(c) After solving Eq. (2), the high-dimensional vectors $\{s^h\}$ are then fed into a trained chart decoder to map them back to charts in Vega-Lite, however, the chart decoder only generates Vega-Lite with the common tokens `NUM` and `STR`.

(d) For each generated Vega-Lite, kNN in the high-dimensional space is used to extract data variables from neighboring charts within a threshold distance $\hat{d}$ and sort them decreasingly by frequency. From the top of the sorted list, the common tokens are replaced with the actual data variables, i.e., a quantitative field for `NUM` and a categorical/ordinal field for `STR`. If there are not enough proper variables to fill in, new, unexplored variables from the dataset are selected. This may represent instances when the analyst selects an empty area with very few charts created.

(e) Finally, an extra check is performed to determine if the generated specifications are valid Vega-Lite. The invalid specifications are then removed and ChartSeer goes back to **(a)** to get more recommendations until a predefined number of charts, i.e., n in **(a)**, are obtained. Next, the recommended charts are ranked using the method in Voyager [65] based on expressiveness and other criteria [42], and then the first m ($m < n$) charts are selected. The rankings from the traditional chart recommendation method are only utilized to enumerate the charts within a vast design space. This enumeration is achieved through steps **(a)-(d)** by involving guidance from the user, resulting in a more efficient recommendation process.

5.2.2 Implementation Notes

There are a number of parameters that were used in the above recommendation process. Within this implementation, The number of samples n was set to $3 \times m$ (where m is determined by the analyst and defaulted to 5) and the sampling radius r is set to 4% of the view size. A k of 5 was used in the kNN search in **(d)** and the threshold distance $\hat{d}$ was set to 0.3 of the maximum distance between existing charts. To solve the optimization problem in **(b)**, BFGS [17] was used with a parallel process that evaluated different

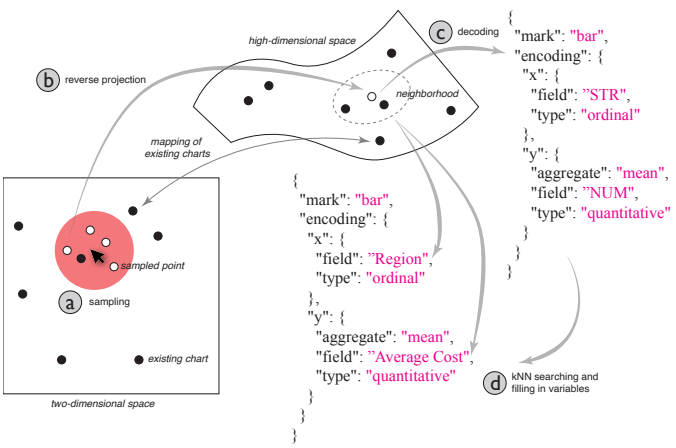


Fig. 7. The chart recommendation process in ChartSeer.

starting conditions to arrive at the best set of conditions. For ranking the recommendations in **(e)**, the `rank` method implemented in CompassQL [3] was used, after removing duplicated charts that were automatically generated.

Alternatively, an encoder and a decoder could be trained with GVAE to directly map charts to and from 2D vectors. Therefore, the dimensionality reduction would be omitted, and thus **(b)** and **(c)** would not be needed. This seems more straightforward, but the accuracy of the encoder and decoder was relatively low based on the results (Table 1). Thus, it is an open question to determine which approach would work best in practice.

5.3 Chart Inspection

ChartSeer allows an analyst to edit the charts they have created that are recommended by the system (**D4**). Once a chart is selected, the Inspection Panel displays the chart along with its Vega-Lite (Figure 5a). To manipulate a chart, expert users can directly edit the Vega-Lite code, whereas novices can use the drop-down controls. Next, an analyst can preview the edited chart and choose to update the chart or add it as a new chart, which will not replace the original chart. Then, the Summarization Panel will update to provide a new overview based on the updated set of charts. This allows analysts to advance their EVA by not only modifying the existing charts, but also refining the recommendations. This encourages a seamless integration of human agency and machine intelligence.

6 Case Study

To learn how ChartSeer can be used within an analyst's workflow, a one-hour interview session was conducted with a data scientist (DS) who had five years of work experience at a large company. As part of his daily job, he used ggplot2 [4], Matplotlib [5], and Vega-Lite [53] to perform EVA.

The goal of this case study was to assess the effectiveness and usefulness of ChartSeer in asynchronous collaborative EVA as it is frequently encountered in practice. This would enable the research team to more efficiently evaluate novel system features in the future. The interview started with a brief tutorial of ChartSeer. The DS was then given the results of some initial analysis of US College data [2] and asked to continue the EVA. The think-aloud protocol was employed

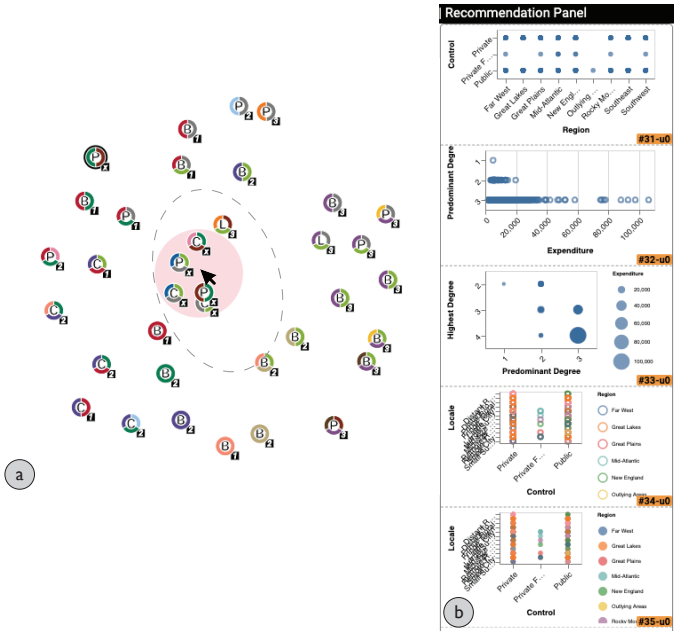


Fig. 8. A data scientist obtaining chart recommendations by clicking on the center of the Summarization View.

during his exploration. We helped resolve any technical questions about the tool. At the termination of the session, additional feedback was collected from the DS. The initial analysis was done by three other data scientists (referred as U1, 2, and 3; all with more than one year of experience) from the same company. The data scientists were asked to perform EVA using the data by creating at least five charts. Each data scientist also used a Jupyter Notebook. The charts were then reviewed and recreated by the research team in Vega-Lite, resulting in 30 charts that were used.

Understanding the prior analysis. After loading the dataset, the DS browsed the Data Table View and noted that there were 18 variables provided about each college (e.g., `Region`, `Cost`, and `Admission Rate`) (Figure 5c). He then utilized the Summarization View to quickly get a big picture of the chats created by other analysts that he had never communicated with (Figure 5b). At a glance, he said that U1 and U2's results seemed interleaved with each other, whereas U3's analysis tended to focus on other aspects, i.e., *"U1 and U2's charts scatter around, but U3's charts form a more coherent cluster on the right,"*.

Next, the DS sampled a few charts to examine by hovering over or clicking on them. Together, with the help of the glyph-based representation of charts, he confirmed his above observation about the prior analysis. He noted that U1 seemed to be interested in the finance of institutions, exploring relations among `Faculty Salary`, `Expenditure`, etc., as well as distributions over `Region` and `Control`. He mentioned that U2 seemed to be more interested in students' finances, exploring variables such as `Cost`, `Debit`, and `Earnings`. He added: *"U1 and U2 seemed both interested in money matters. Their exploration has several overlapping variables. This explains why their charts are mixed together."* The DS commented that U3's exploration focused on the academic aspects, plotting variables such as `Admission Rate`, `SAT`, `ACT`, etc. *"U3's analysis was quite diverging from the other two. This is clearly reflected by the layout of the charts."*

He also noted that from the order of charts in the Chart List View (Figure 5d), the complexity of the charts generally increasing over time for each analyst. He suggested that adding this information to the Summarization View would be helpful for revealing temporal analysis trends.

Identify future opportunities. The DS then wished to continue analyzing this dataset based on the current results. He noticed a couple of empty areas in the Summarization View, which displayed the current landscape of the analysis. For example, one was at the bottom surrounded by mostly U2's charts,, whereas another was in the center between U3's results and the other results (i.e., the dashed circles in Figure 5b). The DS appreciated the Summarization View for revealing these "holes", which he could leverage to guide his future analysis.

The DS was particularly interested in the empty area in the center: *"It is interesting to connect the finance health of institutions and students with the academic aspects."* Thus, he clicked a few spots in that area to get on-demand chart recommendations. The system then provided some charts related to `Highest Degree`, `Predominant Degree`, and `Expenditure` (Figure 5e). He then examined a few recommended charts, commenting that ("they provide a very good basis for me to start.")

Continue the analysis. The DS selected the second recommended chart, a point plot showing the distribution of `Expenditure` over different `Predominant Degrees` (Figure 5e). He observed that the distribution of `Expenditure` becomes more spread out for higher `Predominant Degrees`, with a few institutions having very high `Expenditures` (i.e., over $80,000). He then added the chart directly without modification, and ChartSeer then updated the Summarization View to include this chart and present a new analysis landscape. The DS then hid the background annotations to avoid seeing too much information (Figure 8a). The DS appreciated this feature, i.e., *"It reduces a lot of effort in exploring the charting options."* However, he commented that changing the chart layout after an update caused him difficulty because he could not match his mental model to the current view state.

The DS continued asking for recommendations after observing the updated visual summarization of charts, as there were still *"unexplored areas in the center"* (i.e., the dashed circle in Figure 8a). He said that the recommendations inspired him with a diverse set of charts, in both visual representation and variable inclusion. He was particularly interested in the fourth and fifth charts: *"I never thought about forming a matrix between `Locale/Control` and `Region`, and mapping colors to other variables"* (Figure 8b). He thus selected the fifth one and used the Inspection Panel to augment the chart by adding `Admission Rate` to the color channel. He said that he could then continue the analysis without tediously creating every chart from scratch, commenting: *"Besides reducing the effort, the recommendations encouraged my data exploration and speeded it up. It also extended my analysis vocabulary and coverage."*

7 Controlled User Study

A controlled study was also conducted to assess the novel features of ChartSeer and compare it with a *Baseline* similar

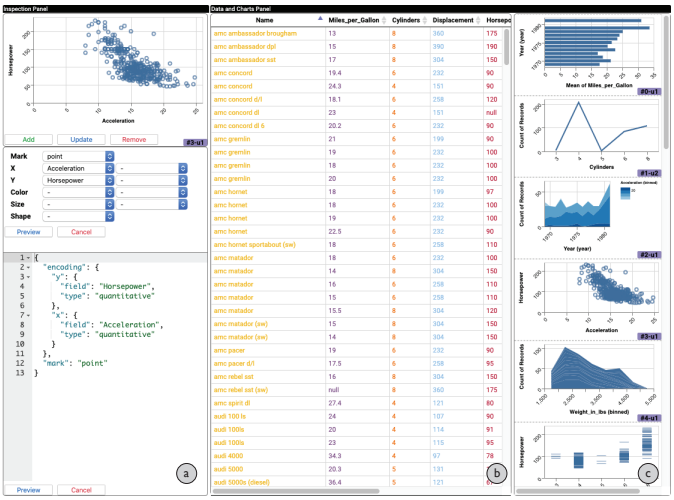


Fig. 9. The Baseline user interface consisted of a: (a) Chart Inspection Panel, (b) Data Table View, and (c) Chart List View, with similar functionalities to ChartSeer's related panels.

to a basic EVA system (i.e., with the functionalities of browsing the dataset and creating charts; Figure 9). The following study questions were of interest: **(S1)** what are the effects on the outcomes of participants' analyses, **(S2)** what are participants' attitudes towards the new features, and **(S3)** how do participants use these features?

7.1 Participants and Apparatus

Twenty-four participants (i.e., 13 females and 11 males, aged 19–35) were recruited to participate in the study via university social networks, including 17 graduate students, 4 undergraduate students, and 3 professional analysts. Their backgrounds ranged from science, to commerce and engineering, with an average of 3.7 ($\sigma = 2.7$) years of experience in data analysis. Their self-reported familiarity of using off-the-shelf tools (e.g., Excel, ggplot [4], Maplotlib [5]) to create charts was: $M = 3, Mo = 3$ (i.e., 1–not familiar at all, 5–extremely familiar). Participants completed the study using a desktop computer with a 27-inch monitor and a mouse & keyboard. The application window sizes of ChartSeer and Baseline were fixed across the study.

7.2 Design, Tasks, and Procedure

A between-subjects design was used with 12 participants in each condition (i.e., ChartSeer and Baseline). To begin, each participant was given a short tutorial about the study system. Participants were put in a scenario representing a multi-session EVA, i.e., continuing an analysis started by another individual. This allowed for a more realistic evaluation of the chart summarization and recommendations in ChartSeer. Another smaller dataset based on the Cars data [1], was prepared using the same method in that described in the case study. Thus, participants completed one practice session using the Cars dataset and one study session using the Colleges dataset.

There were two phases during the study, i.e., a *Summarization* phase and an *Exploration* phase. First, participants were asked to browse the existing charts and try to understand and synthesize the results. They were not allowed to edit the charts, create new ones, or use the recommendation (in the ChartSeer condition). A short interview was then conducted to collect their findings and thoughts about the system. Second, participants were instructed to continue the EVA by further exploring the data, creating new charts, and discovering new insights. A second interview was then conducted to collect their feedback about the system. Splitting the task into two phases allowed for relatively independent feedback and observations about the chart summarization and recommendation features to be obtained, as well as ensure that participants were able to adequately pick up the prior analysis before continuing on their own.

Participants completed the tasks with time limits. During the practice trial, six minutes was given for Summarization and eight minutes for Exploration. In the non-practice trial, 10 minutes was given for Summarization and 15 minutes was given for Exploration. At the end, participants completed a questionnaire based on 7-point Likert Scales about their experiences when using the system. All the interview sessions were audio-recorded, and participants' interactions with the system were logged. The study lasted about 80 minutes and participants received $25 in compensation.

7.3 Results and Analysis

7.3.1 Effect on Task Outcomes

To understand how ChartSeer would affect the results of participants' analysis (S1), the number of new charts *added* and the number of charts *updated* (or replaced) was computed. These measures were used because they reflect two different user goals. The analysis is reported with 95% confidence intervals (in subscripts) using the bootstrap method, effect sizes using Cohen's d, and non-parametric Mann-Whitney tests, following the statistical methods in [16].

A slight tendency to ***add more charts using ChartSeer was found compared to the Baseline, whereas more charts were found to be updated when using the Baseline compared to ChartSeer*** (Figure 10a,f). This may have been because the creation of charts requires more effort when recommendations are not provided.

The two heatmaps in Figure 10 show the frequency of each data variable and each encoding channel used by each participant, respectively. The usage patterns were quantified using two metrics, i.e., *coverage* and *uniqueness*. Coverage is a commonly accepted measure when evaluating EVA (e.g., [51]), where the number of distinct occurrences are counted. Higher coverage indicates that a tool encourages user behaviors that touch on a *broader* scope of the measured aspect. Uniqueness is a new metric introduced in [20], which captures the diverse engagement of EVA, i.e., how different a user's exploration is compared to the explorations of others. Higher uniqueness indicates that a tool encourages more *heterogeneous* behaviors of a measured aspect.

ChartSeer was found to ***lead to a broader range of data variables and visual encodings***. That is, the added charts in ChartSeer had higher coverage, i.e., with more distinct data variables (ChartSeer: $\mu = 7.08_{[6.08,8.08]}$; Baseline: $\mu = 4.67_{[3,6.42]}$; effect size: $d = 0.92_{[0.03,1.85]}$) as well as more distinct visual channels (ChartSeer: $\mu = 3.83_{[3.17,4.25]}$; Baseline: $\mu = 2.42_{[1.53,3]}$; effect size: $d = 1.24_{[0.42,1.98]}$; Figure 10b,c). This can also be observed from the heatmaps, where there are more colored cells in ChartSeer. The reason

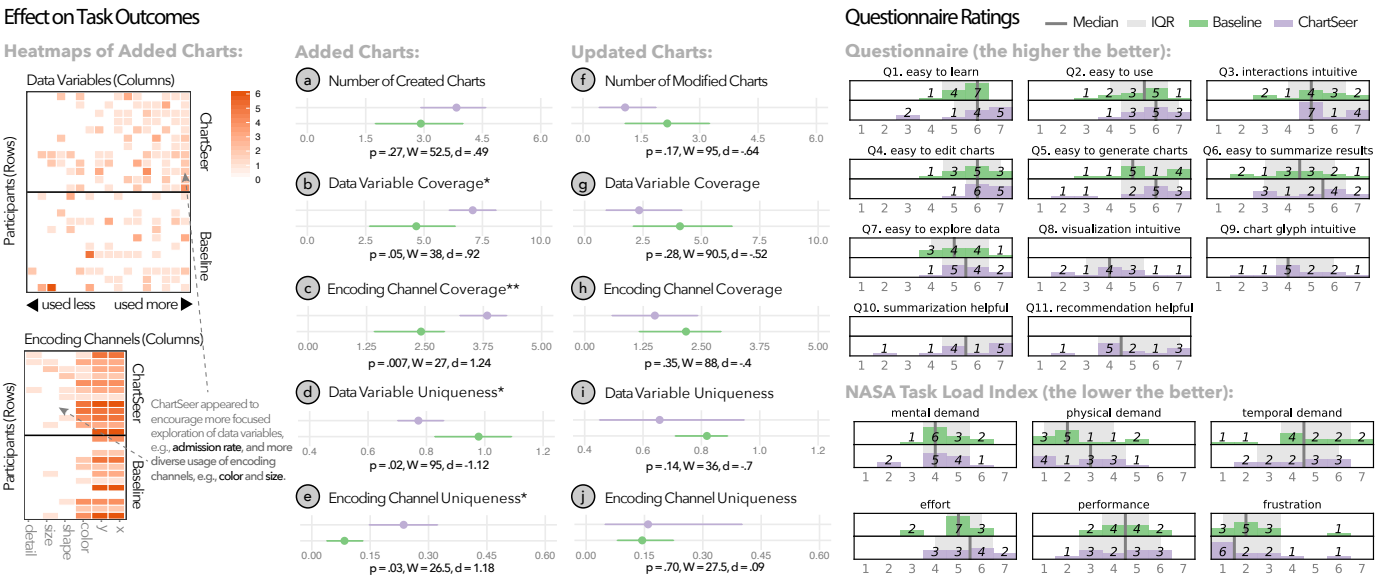


Fig. 10. Analysis of participants' outcomes (left) and questionnaire ratings (right). For each measure, group means with 95% CIs are reported, along with the results of a Mann-Whitney test (* and ** denote $p < .05$ and $.01$, respectively).

may be that participants were inspired and encouraged to using more expressive charts based on the recommendations, which could contain more distinct variables and channels than those in charts that they usually create.

Second, ChartSeer ***encouraged a more focused exploration of data variables***, exhibiting a lower uniqueness score (ChartSeer: $\mu = 0.77_{[0.7,0.86]}$; Baseline: $\mu = 0.98_{[0.82,1.1]}$; effect size: $d = -1.12_{[-2.22,0.03]}$; Figure 10d). Also, the upper heatmap demonstrated that more variables were commonly and frequently used by participants in ChartSeer (e.g., `Admission Rate`). This may have been because participants tended to explore similar open areas that were indicated by the visual summarization of charts and requested recommendations in those areas. ChartSeer would then have provided charts with data variables that were influenced by the existing close-by charts in local regions so participants' explorations may have become more uniform. In the Baseline condition, where there was no meta-visualization, it is possible that participants were unaware of these open areas and thus investigated variables in a more random manner.

Third, when using ChartSeer, participants ***explored the data from more heterogeneous visual perspectives***, i.e., adding charts with more diverse encodings (Figure 10e). Particularly, the recommendations encouraged some participants to use novel encoding channels, such as Color and Shape, compared to others. As shown in the lower heatmap in Figure 10, the combinatorial patterns of colored cells were more diverse in ChartSeer, resulting in ChartSeer having higher uniqueness (ChartSeer: $\mu = 1.18_{[0.37,2.14]}$; Baseline: $\mu = 0.08_{[0.04,0.13]}$; effect size: $d = 1.18_{[0.37,2.14]}$).

The usage of data variables and encoding channels within updated charts was also evaluated, however, no significant results were found. This indicates that participants performed similarly in both conditions. However, the updated charts are less of an indicator of the effects of ChartSeer because the goal is to encourage continuing EVA with new knowledge rather than fixating on old charts. Future studies are needed to investigate this further, however.

7.3.2 *Usage, Benefits, and Challenges*

To answer S2 and S3, participants' ratings about ChartSeer and Baseline were compared, in addition to transcriptions of the interviews, to understand ChartSeer's usage and its limitations.

Questionnaire Ratings. A Mann-Whitney evaluation of the NASA Task Load Index (TLX) [6] results found no significant difference between the two conditions (Figure 10), however there appeared to be a trend towards participants perceiving that ChartSeer demanded more physical effort. Although the interface of ChartSeer is more complicated, within the Likert-scale questionnaire data, participants did not perceive a difficulty when learning or using the system compared to the Baseline interface (Q1–7). Although not significant, ChartSeer seemed to be easier for participants to use to summarize and generate charts during their analysis (Q5–6). Participants also tended to positively accept the new visualization and features of ChartSeer that were not part of the Baseline condition (Q8–11), although these results were also not significant.

Feedback on Chart Summarization. The Summarization Panel effectively facilitated participants' EVA in three main ways. The first was to help them ***understand the current analysis landscape***. They appreciated that charts having similar variables and encodings were *"close together"* because this enabled them to see what had been explored and how. Moreover, it encouraged them to *"investigate other variables and correlations not shown in the Chart List View"–p2* Secondly, it helped them ***gain inspirations for further exploration***. Due to the numerous possible combinations of variables and encodings, the Summarization View *"could indicate an area to focus on at the beginning"–p17*. They mentioned that the existing charts served as a source of inspiration to help identify sets of variables for future analysis. Lastly, participants leveraged the relative distances between the generated and existing charts to ***assess the quality of their created charts***: *"I saw my charts were quite far away from other charts, so I thought I did a good job at finding some new directions that hadn't been explored before"–p20*.

Feedback on Chart Recommendation. Participants appreciated the chart recommendations in ChartSeer and used them *"to explore the variables that have not been shown in the existing charts"–p2*. They also said that although it might be relatively easy to locate the variables that have not been explored, it is often difficult *"to determine a combination of variables to put together in one chart, whereas the system automatically generates new combinations for you"–p18*.

Participants triggered the chart recommendation in three ways. First, they requested recommendations in ***globally empty areas in the Summarization View***, which were often viewed as unexplored areas, e.g., *"I clicked on the empty space, and chose one recommended chart that made sense to me"–p19*. Moreover, participants explored ***locally empty areas around existing charts*** because they felt that starting from a local region would better guide their exploration, e.g., *"I always start from an area near an existing chart that I'm interested in, because it helped me start from somewhere that makes sense to me. If I go to an empty space out of nowhere, I would have no idea what variables would be introduced in the recommendation and why those charts are recommended to me."–p9*. After generating a new chart, some participants used it as a new *"anchor point"* and clicked on its surrounding area to continue exploring this area for more recommendations. Further, participants used chart recommendations ***with the guidance from chart clusters***. Charts with similar variables and visual encodings tended to be close to each other in the Summarization View. Such information helped determine whether they should continue exploring the same region to uncover the relationships of existing variables or if they should move on to the next region to explore completely new variables, e.g., *"I envisioned the panel as a spectrum of relationships between variables. I started from the purple- and gray-ish area (i.e., `Median Admission Rate` and `Control`) and worked progressively into its opposite side, which is the pink- and green-ish area (i.e., `Highest Degree` and `Expenditure`). I wanted to identify holes and close them in each area."–p19*.

However, ChartSeer sometimes recommended charts containing three or more variables and several participants felt that such ***charts were somewhat complicated***, e.g., *"Simple charts with two factors are easy to understand and easy to see trends"–p20*. Another challenge was that ***some recommended charts were unexpected***. This often happened when participants clicked on a locally empty area near an existing chart and expected to see charts containing the same variables instead of new ones. Several participants also mentioned that ***the system could not fully understand their intent***, e.g., *"If the system can ask me what are the variables that I'm mostly curious about and then suggest charts about those variables. Then when I click on the empty region, the system only suggests me the ones with a variable that I'm interested in."–p20*.

8 Discussion

The results of the case study and user evaluation suggested that ChartSeer has advantages and can help steer EVAs by combining the efforts of human knowledge and machine intelligence. The case study revealed that ChartSeer was helpful for understanding the overall state of an analysis, as well as providing useful information and suggestions to advance the EVA in a large analysis space. The controlled study suggests that ChartSeer encouraged participants to create more charts with more coverage and visual diversity, thereby leading them towards more effective EVA. Moreover, participants could better steer and control their EVA by communicating with ChartSeer by pointing at certain locations in the view and having a more focused exploration of variables. Although it has many benefits ChartSeer does not come without limitations.

First, several participants were puzzled about the actual meanings of the dimensions in the Summarization View, although they appreciated the visualization. Technically, the two dimensions have no physical meaning as they are produced by the dimensionality reduction. Future work should explore ways to help analysts conceptualize and manipulate such dimensions, perhaps employing AxiSketcher [38] techniques or forward and backward projection [12]. Also, participants pointed out that ChartSeer does not visualize the temporal information of a prior analysis, which could be useful for one to further understand the EVA process. Various strategies from the literature could be applied to support such temporal behavior understanding [7], [70].

Second, the controlled study revealed that participants sometimes expected ChartSeer to generate charts that satisfied specific constraints (e.g., containing certain variables). However, it is currently difficult for analysts to communicate such specific intentions, even though they could indicate where in the space they wanted a recommendation. Designing semantic user interactions [28] could be necessary to allow for better steering of EVA. As there may be a limited number of useful recommendations within a small subspace of the analysis landscape, designing visual cues and mechanisms for guiding analysts to think "out of the box" would be useful to investigate in the future.

Third, there is still room to improve the recommendation model even though ChartSeer was able to achieve much higher reconstruction accuracy with GVAE [37] compared to the literature. One way would be to collect more training data, as the performance of deep learning models tends to rely on the size of the training dataset. Another way would be to integrate traditional rule/heuristic methods (e.g., [43], [65]) with this learning-based approach, for example, by filtering the automatically generated recommendations or constraining the generation process at the beginning. Currently, ChartSeer does not provide an initial set of recommendations for analysts to start with. However, a random sample of the analysis space can be utilized to generate some initial charts, which would be especially useful when the space is vast. Some heuristics may need to be applied to guide the recommendation, such as restricting variables that have stronger correlations or leveraging some rule-based methods. Further, the results indicated that participants explored data variables less uniquely when they were provided with recommendations, although they exhibited higher coverage. It would thus be interesting to tune the recommendation process to include more diverse variables, however, more empirical evaluation would be needed to determine if this would produce better EVA results.

The design of the controlled study also has limitations. All participants started at the same point to continue the analysis. It is unknown how ChartSeer would benefit EVA differently in a multi-session process. For example, analysts

joining late in the process may have too many charts to digest and little empty space to explore, giving ChartSeer a larger advantage. Further, gaining knowledge about prior analyses using meta-visualization is an integral part of working with, and employing,the chart recommendations. These two phases in the study were interleaved, however, they are iterative processes in practice. Within the constraints of an in-lab study, only one iteration could be investigated, however, it would be useful to run a long-term deployment study to assess ChartSeer in more practical EVA scenarios. The study compared two conditions, Baseline and ChartSeer (which was equipped with both summarization and recommendation functions). While the case study with the data scientist and the controlled study shed some light on these two novel features of ChartSeer with questionnaires asking participants for their experiences respectively (Figure 10), a 2-by-2 factorial experimental design would allow for an evaluation of these two features independently and thus provide a systematical understanding of their individual effects.

9 Conclusion

This work introduced ChartSeer, a system for steering EVA by integrating human agency and machine automation through the use of interactive visualizations and machine learning. ChartSeer was developed based on design considerations distilled from the literature and leverages deep learning models to enable the meta-visualization of prior analysis and the interactive recommendation of charts to support multi-session asynchronous EVA. The results of a case study and a comparative user study found that ChartSeer led participants to use a broader range of data variables and visual encodings, encouraged a more focused exploration of data variables, led to more heterogeneous visual explorations of data, and encouraged participants to add more charts compared to a baseline.

Acknowledgments

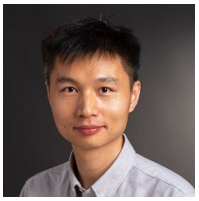
This work is supported in part by an NSERC Discovery Grant and a gift fund from Adobe Systems, Inc.

References

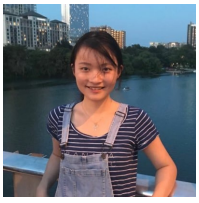
[1] Cars dataset. https://github.com/vega/vega-datasets/blob/master/data/cars.json, accessed in 2020.
[2] Colleges dataset. https://collegescorecard.ed.gov/data, accessed in 2020.
[3] Compassql. https://github.com/vega/compassql, accessed in 2020.
[4] ggplot2. https://ggplot2.tidyverse.org/, accessed in 2020.
[5] Matplotlib. https://matplotlib.org/, accessed in 2020.
[6] NASA Task Load Index (TLX). https://humansystems.arc.nasa.gov/groups/TLX/, accessed in 2020.
[7] B. Bach, C. Shi, N. Heulot, T. Madhyastha, T. Grabowski, and P. Dragicevic. Time curves: Folding time to visualize patterns of temporal evolution in data. *IEEE Trans. on Vis. and Comp. Graphics*, 22(1):559–568, 2015.
[8] L. Battle and J. Heer. Characterizing exploratory visual analysis: A literature review and evaluation of analytic provenance in tableau. *Computer Graphics Forum*, 38(3):145–159, 2019.
[9] F. Bouali, A. Guettala, and G. Venturini. Vizassist: An interactive user assistant for visual data mining. *Vis. Comput.*, 32(11):1447–1463, 2016.
[10] M. Brehmer and T. Munzner. A multi-level typology of abstract visualization tasks. *IEEE Trans. on Vis. and Comp. Graphics*, 19(12):2376–2385, 2013.
[11] S. M. Casner. Task-analytic approach to the automated design of graphic presentations. *ACM Trans. on Graphics*, 10(2):111–151, 1991.
[12] M. Cavallo and Ç. Demiralp. A visual interaction framework for dimensionality reduction based data exploration. In *Extended Abstracts of the CHI Conf. on Human Factors in Computing Systems*, 2018.
[13] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *CoRR*, 2014.
[14] C. Collins, G. Penn, and S. Carpendale. Bubble sets: Revealing set relations with isocontours over existing visualizations. *IEEE Trans. on Vis. and Comp. Graphics*, 15(6):1009–1016, 2009.
[15] K. A. Cook and J. J. Thomas. *Illuminating the path: The research and development agenda for visual analytics*. National Visualization and Analytics Ctr, 2005.
[16] G. Cumming. *Understanding the new statistics: Effect sizes, confidence intervals, and meta-analysis*. Routledge, 2013.
[17] F. E. Curtis and X. Que. A quasi-newton algorithm for nonconvex, nonsmooth optimization with global convergence guarantees. *Mathematical Programming Computation*, 7(4):399–428, 2015.
[18] C. Demiralp, C. E. Scheidegger, G. L. Kindlmann, D. H. Laidlaw, and J. Heer. Visual embedding: A model for visualization. *IEEE Computer Graphics and Applications*, 34(1):10–15, 2014.
[19] V. Dibia and Ç. Demiralp. Data2vis: Automatic generation of data visualizations using sequence to sequence recurrent neural networks. *CoRR*, abs/1804.03126, 2018.
[20] M. Feng, E. Peck, and L. Harrison. Patterns and pace: Quantifying diverse exploration behavior with visualizations on the web. *IEEE Trans. on Vis. and Comp. Graphics*, 25(1):501–511, 2018.
[21] T. Fujiwara, J.-K. Chou, Shilpika, P. Xu, L. Ren, and K.-L. Ma. An incremental dimensionality reduction method for visualizing streaming multidimensional data. *IEEE Trans. on Vis. and Comp. Graphics*, 26(1):418–428, 2020.
[22] O. Gilson, N. Silva, P. Grant, and M. Chen. From web data to visualization via ontology mapping. *Computer Graphics Forum*, 27(3):959–966, 2008.
[23] D. Gotz and Z. Wen. Behavior-driven visualization recommendation. In *Proc. of the Int'l Conf. on Intelligent User Interfaces*, pages 315–324, 2009.
[24] J. Heer, F. B. Viégas, and M. Wattenberg. Voyagers and voyeurs: supporting asynchronous collaborative information visualization. In *Proc. of the CHI Conf. on Human factors in computing systems*, pages 1029–1038. ACM, 2007.
[25] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
[26] E. Horvitz. Principles of mixed-initiative user interfaces. In *Proc. of the CHI Conf. on Human factors in computing systems*, 1999.
[27] K. Z. Hu, M. A. Bakker, S. Li, T. Kraska, and C. A. Hidalgo. Vizml: A machine learning approach to visualization recommendation. *arXiv:1808.04819*, 2018.
[28] X. Hu, L. Bradel, D. Maiti, L. House, C. North, and S. Leman. Semantics of directly manipulating spatializations. *IEEE Trans. on Vis. and Comp. Graphics*, 19(12):2052–2059, 2013.
[29] J. Hullman, S. Drucker, N. H. Riche, B. Lee, D. Fisher, and E. Adar. A deeper understanding of sequence in narrative visualization. *IEEE Trans. on Vis. and Comp. Graphics*, 19(12):2406–2415, 2013.
[30] P. Isenberg, N. Elmqvist, J. Scholtz, D. Cernea, K.-L. Ma, and H. Hagen. Collaborative visualization: definition, challenges, and research agenda. *Information Visualization*, 10(4):310–326, 2011.
[31] P. Isenberg, D. Fisher, M. R. Morris, K. Inkpen, and M. Czerwinski. An exploratory study of co-located collaborative visual analytics around a tabletop display. In *IEEE Sym. on Visual Analytics Science and Technology*, pages 179–186. IEEE, 2010.
[32] T. Jankun-Kelly, K. Ma, and M. Gertz. A model and framework for visualization exploration. *IEEE Trans. on Vis. and Comp. Graphics*, 13(2):357–369, 2007.
[33] H. John. *Introduction to Automata Theory, Languages, and Computation*. 2008.
[34] Y. Kim, K. Wongsuphasawat, J. Hullman, and J. Heer. Graphscape: A model for automated reasoning about visualization similarity and sequencing. In *Proc. of the CHI Conf. on Human Factors in Computing Systems*, pages 2628–2638. ACM, 2017.

[35] D. Kingma and M. Welling. Auto-encoding variational bayes. In *Proc. of the Int'l Conf. on Learning Representations*, 2014.
[36] J. B. Kruskal and M. Wish. *Multidimensional scaling*, volume 11. Sage, 1978.
[37] M. J. Kusner, B. Paige, and J. M. Hernández-Lobato. Grammar variational autoencoder. In D. Precup and Y. W. Teh, editors, *Proc. of the 34th Int'l Conf. on Machine Learning*, volume 70, pages 1945–1954, 2017.
[38] B. C. Kwon, H. Kim, E. Wall, J. Choo, H. Park, and A. Endert. Axisketcher: Interactive nonlinear axis mapping of visualizations through user drawings. *IEEE Trans. on Vis. and Comp. Graphics*, 23(1):221–230, 2016.
[39] Y. Luo, X. Qin, N. Tang, and G. Li. DeepEye: Towards automatic data visualization. In *Proc. of the Int'l Conf. on Data Engineering*, 2018.
[40] K.-L. Ma. Image graphs-a novel approach to visual data exploration. In *Proc. of the Conf. on Visualization*, pages 81–88. IEEE Computer Society Press, 1999.
[41] L. v. d. Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.
[42] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Trans. on Graphics*, 5(2):110–141, 1986.
[43] J. Mackinlay, P. Hanrahan, and C. Stolte. Show me: Automatic presentation for visual analysis. *IEEE Trans. on Vis. and Comp. Graphics*, 13(6):1137–1144, 2007.
[44] N. Mahyar and M. Tory. Supporting communication and coordination in collaborative sensemaking. *IEEE Trans. on Vis. and Comp. Graphics*, 20(12):1633–1642, 2014.
[45] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv:1301.3781*, 2013.
[46] D. Moritz, C. Wang, G. L. Nelson, H. Lin, A. M. Smith, B. Howe, and J. Heer. Formalizing visualization design knowledge as constraints: Actionable and extensible models in draco. *IEEE Trans. on Vis. and Comp. Graphics*, 25(1):438–448, 2019.
[47] J. Peltonen and Z. Lin. Information retrieval approach to meta-visualization. *Machine Learning*, 99(2):189–229, 2015.
[48] P. Pirolli and S. Card. Information foraging. *Psychological Review*, 106(4):643–675, 1999.
[49] J. Poco and J. Heer. Reverse-engineering visualizations: Recovering visual encodings from chart images. *Computer Graphics Forum*, 36(3):353–363, 2017.
[50] S. F. Roth, J. Kolojejchick, J. Mattis, and J. Goldstein. Interactive graphic design using automatic presentation knowledge. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 112–117, 1994.
[51] A. Sarvghad and M. Tory. Exploiting analysis history to support collaborative data analysis. In *Proc. of Graphics Interface Conf.*, pages 123–130. Canadian Information Processing Society, 2015.
[52] A. Sarvghad, M. Tory, and N. Mahyar. Visualizing dimension coverage to support exploratory analysis. *IEEE Trans. on Vis. and Comp. Graphics*, 23(1):21–30, 2017.
[53] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-lite: A grammar of interactive graphics. *IEEE Trans. on Vis. and Comp. Graphics*, 23(1):341–350, 2017.
[54] M. Sedlmair, C. Heinzl, S. Bruckner, H. Piringer, and T. Moller. Visual parameter space analysis: A conceptual framework. *IEEE Trans. on Vis. and Comp. Graphics*, 20(12):2161–2170, 2014.
[55] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.
[56] M. Tobiasz, P. Isenberg, and S. Carpendale. Lark: Coordinating co-located collaboration with information visualization. *IEEE Trans. on Vis. and Comp. Graphics*, 15(6):1065–1072, 2009.
[57] J. W. Tukey. *Exploratory Data Analysis*. Pearson, 1977.
[58] M. Vartak, S. Madden, A. Parameswaran, and N. Polyzotis. Seedb: Automatically generating query visualizations. *Proc. of VLDB Endowment*, 7(13):1581–1584, 2014.
[59] F. B. Viegas, M. Wattenberg, F. Van Ham, J. Kriss, and M. McKeon. Manyeyes: a site for visualization at internet scale. *IEEE Trans. on Vis. and Comp. Graphics*, 13(6), 2007.
[60] M. Voigt, S. Pietschmann, and L. Grammel. Context-aware recommendation of visualization components. In *Proc. of the Int'l Conf. on Information, Process, and Knowledge Management*, 2012.
[61] M. Wattenberg and J. Kriss. Designing for social data analysis. *IEEE Trans. on Vis. and Comp. Graphics*, 12(4):549–557, 2006.
[62] W. Willett, J. Heer, J. Hellerstein, and M. Agrawala. CommentSpace: Structured support for collaborative visual analysis. In *Proc. of the Conf. on Human Factors in Computing Systems (CHI)*, pages 3131–3140. ACM, 2011.
[63] I. H. Witten, E. Frank, and M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 3rd edition, 2011.
[64] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Towards a general-purpose query language for visualization recommendation. In *Proc. of the Workshop on Human-In-the-Loop Data Analytics*, pages 4:1–4:6, 2016.
[65] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Trans. on Vis. and Comp. Graphics*, 22(1):649–658, 2016.
[66] K. Wongsuphasawat, Z. Qu, D. Moritz, R. Chang, F. Ouk, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager 2: Augmenting visual analysis with partial view specifications. In *Proc. of the CHI Conf. on Human Factors in Computing Systems*, pages 2648–2659, 2017.
[67] S. Xu, C. Bryan, J. K. Li, J. Zhao, and K.-L. Ma. Chart constellations: Effective chart summarization for collaborative and multi-user analyses. *Computer Graphics Forum*, 37(3):75–86, 2018.
[68] J. Zhao, C. Bhatt, M. Cooper, and D. A. Shamma. Flexible learning with semantic visual exploration and sequence-based recommendation of MOOC videos. In *Proc. of the CHI Conf. on Human Factors in Computing Systems*, 2018.
[69] J. Zhao, M. Glueck, S. Breslav, F. Chevalier, and A. Khan. Annotation graphs: A graph-based visualization for meta-analysis of data based on user-authored annotations. *IEEE Trans. on Vis. and Comp. Graphics*, 23(1):261–270, 2017.
[70] J. Zhao, M. Glueck, P. Isenberg, F. Chevalier, and A. Khan. Supporting handoff in asynchronous collaborative sensemaking using knowledge-transfer graphs. *IEEE Trans. on Vis. and Comp. Graphics*, 24(1):340–350, 2017.

Jian Zhao is an assistant professor in the Cheriton School of Computer Science at the University of Waterloo. His research interests include Information Visualization, Human-Computer Interaction, Visual Analytics, and Data Science. His work contributes to the development of advanced interactive visualizations that promote the interplay of humans, machines, and data.



Mingming Fan is an assistant professor in the School of Information at the Golisano College of Computing and Information Sciences at the Rochester Institute of Technology. He research falls at the intersection of Human-Computer Interaction and Artificial Intelligence, focusing on User Experience- and Accessibility-related challenges and the design of AI-empowered interactive systems to address these challenges.



Mi Feng is a data visualization scientist at Twitter Inc. She is interested in leveraging data visualization techniques to understand our minds and behaviors in digital worlds. These behaviors include, but are not limited to, our interactions with digital interfaces such as websites, mobile apps, data visualizations, VR environments, and so on.