An Automated Approach to Reasoning About Task-Oriented Insights in Responsive Visualization

Hyeok Kim, Ryan Rossi, Abhraneel Sarma, Dominik Moritz, and Jessica Hullman

Abstract— Authors often transform a large screen visualization for smaller displays through rescaling, aggregation and other techniques when creating visualizations for both desktop and mobile devices (i.e., responsive visualization). However, transformations can alter relationships or patterns implied by the large screen view, requiring authors to reason carefully about what information to preserve while adjusting their design for the smaller display. We propose an automated approach to approximating the loss of support for task-oriented visualization insights (identification, comparison, and trend) in responsive transformation of a source visualization. We operationalize identification, comparison, and trend loss as objective functions calculated by comparing properties of the rendered source visualization to each realized target (small screen) visualization. To evaluate the utility of our approach, we train machine learning models on human ranked small screen alternative visualizations across a set of source visualizations. We find that our approach achieves an accuracy of 84% (random forest model) in ranking visualizations. We demonstrate this approach in a prototype responsive visualization recommender that enumerates responsive transformations using Answer Set Programming and evaluates the preservation of task-oriented insights using our loss measures. We discuss implications of our approach for the development of automated and semi-automated responsive visualization recommendation.

Index Terms—Task-oriented insight preservation, responsive visualization

1 Introduction

Visualization authors often transform their designs to accommodate different audience, styles, or display types. For example, authors might simplify charts for audiences with different graphical literacy, altering information conveyed in the new design [14]. Authors of responsive visualizations create multiple designs for different screen sizes and interactivity [30, 43]. However, transforming visualizations may invoke trade-offs between desirable design criteria. For instance, in Fig. 1, proportionate rescaling (a) makes it harder to compare values along the y-axis due to the reduced absolute height, while increasing the relative height (b) or transposing (c) can distort the shape of the represented distribution. Increasing the bin size (d) reduces possible comparisons and a viewer's ability to see distributional detail.

Unfortunately, design trade-offs make it difficult to reason about preserving takeaways or "insights" under visualization design transformations. Designers may need to iteratively try out different combinations of strategies like those in Fig. 1 and compare them with the original design [30]. Kim et al. [43] identify trade-offs in designing responsive visualization where authors try to strike a balance between maintaining graphical density (i.e., an appropriate number of marks per unit screen size) and the preservation of a user's ability to arrive at certain insights. For example, authors need to decide either to preserve distributional characteristics with higher visual density by rescaling proportionately (a) or to adjust visual density while losing distributional details by changing the bin size (d). Currently these decisions are made manually.

We contribute an automated approach to approximating the amount of change to task-oriented insights—insights that viewers are likely to be able to obtain from a visualization by performing visual tasks—under design transformation. We define measures that approximate a visualization's support for three low-level visual analytic tasks discussed in the literature [2, 7]: the viewer's ability to identify a datum, to compare pairs of data points, and to perceive a multivariate trend. We demonstrate the use of our measures for choosing between alternative transformations of a source large screen visualization in a responsive design context. We provide a prototype automated design recommender for responsive visualization that enumerates responsive design transformations based on an input source view and reasons about changes in task-oriented insights from the source design to each transformed design. Our recommender supports scatterplots, bar charts, line graphs, and heatmaps with position, color, size, and shape encoding channels.

We train and test different machine learning (ML) models based on our measures to evaluate their utility for automatically ranking small screen visualization design alternatives given a large screen view. We achieve up to 84% accuracy (via a random forest model) in ranking a set of responsive transformations across a set of six source large screen views spanning different encoding channels. Models trained with our measures outperform two baseline models based on simple heuristics related to chart size changes (59%) and transposing of axes (63%). We discuss implications of our work for future research, including recommender-driven responsive visualization authoring and generalizing our approach to further visualization design domains such as simplification and style transfer.

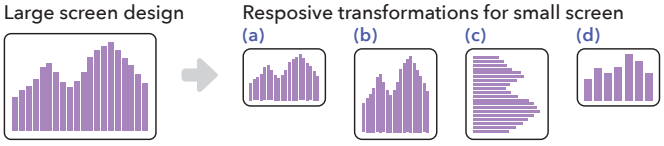


Fig. 1. Example responsive transformations for small screen generated from a large screen design: (a) proportionate rescaling, (b) disproportionate rescaling, (c) transposing axes, and (d) increasing bin size.

2 Related Work

2.1 Responsive Visualization and Design Transformations

We use *visualization design transformation* to refer to the transformation of a source visualization specification to a new visualization specification intended to better achieve certain context-specific constraints. These might be screen size limitations for responsive visualization, audience-related constraints in visualization simplification or audience retargeting (e.g., [5, 40]), or style constraints in style transfer [24, 25], for example. Visualization design transformation differs from creating

- *Hyeok Kim, Abhraneel Sarma, and Jessica Hullman are with Northwestern University. E-mail: hyeokkim2024@u.northwestern.edu, abhraneelsarma2024 @u.northwestern.edu, jhullman@northwestern.edu.*
- *Ryan Rossi is with Adobe Research. E-mail: ryrossi@adobe.com.*
- *Dominik Moritz is with Carnegie Mellon University. E-mail: domoritz@cmu.edu.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

multiple different views from the same dataset (e.g., a visualization sequence or dashboard) in that transformations of an original source view typically are intended to preserve many properties of the source while changing select properties.

Prior work on responsive visualization, which tends to focus on Web-based communicative visualization, or scalable visualization [13] more broadly, emphasizes the importance of maintaining intended takeaways between source and transformed views. Analyzing 378 responsive visualization pairs on desktop and mobile devices, Kim et al. [43] identify density-message trade-offs in responsive visualization where authors need to balance adjusting visual density or complexity for different screen types while maintaining patterns, trends or other important information conveyed in the source view. Focusing on maintaining key information at different scales, earlier work on visualization resizing introduces algorithms that repeatedly remove the pixels determined to be least important [18] and iteratively minimize scaling in more salient regions [76], for example. We extend prior approaches by proposing approximation methods for task-oriented visualization insights.

Defaulting to simpler views over complex, over-encoded plots is often recommended when exploring or publicizing complex data [41]. Authors accomplish this through data-level transformations, such as data abstraction, clutter reduction, filtering, or clustering. For example, data abstraction studies have attempted to enhance the simplicity of a view while preserving original structure or insights (e.g., aggregating a large movement dataset [1], using interactive dimensionality reduction [39], using hierarchical aggregation [19], and measuring the quality of an abstraction [14]).

2.2 Visualization Recommendation

We discuss two approaches in visualization recommendation—insight-based and similarity-based—that are relevant to our goal of approximating changes in task-oriented insights. Prior work on visualization recommendation employs statistical calculations to characterize properties of a visualization thought to relate to the insights a user can draw from it. Often these 'insights' are intended to capture how well a user can perform analytic tasks, such as recognizing trends or identifying and comparing data points. Tang et al. [65] suggest detecting 'top-k insights' from data using statistical significance testing (e.g., low p-value of a linear regression coefficient for slope insight). Similarly, Foresight [17], DataSite [15], and Voder [61] use statistics calculated on the data, such as correlation coefficient and interquartile range, and recommend visualization types predicted to better support extracting such information. However, statistics on data are invariant for views sharing the same data set and hence of limited use for comparing different ways of visualizing the same underlying data. Our work instead considers statistics calculated on the rendered visualization.

Several prior visualization recommenders model similarity between views, but assume a scenario where the underlying dataset changes. GraphScape [45] offers a view similarity model that assigns costs to visualization pairs that are intended to approximate the cognitive cost of transitioning from one view to another in a visualization sequence. GraphScape applies an a priori cost model in which data transformation (e.g., binning, modifying scales) is always less costly than changes in encoding. Hence, filtering data has a lower cost than transposing axes. However, filtering operations like removing a bar from a bar chart or rescaling a y-axis can significantly change the presumed "takeaways" of a chart (e.g., [23, 31]). The space of transformations covered by GraphScape also does not include view size transformations, so it cannot assign costs to changes in aspect ratio common to responsive visualization. Although Dziban [47] extends GraphScape to suggest a view that is 'anchored' to the previous view for an exploratory data analysis process, it also assumes different subsets of data between the previous and current views and focuses more on similar chart encodings than on preserving task-oriented insights.

2.3 Comparing Visual Structure by Processing Signal

Signal processing-based approaches analyze the underlying visual or perceptual structure of a visualization to enable multi-scale visualizations (i.e., providing different insights at different scales) and to enhance visualization effectiveness. Prior work has attempted to enable multi-scale views through perceptual organization analysis of a information graphic at each scale [72, 73] and hybrid-image visualization that displays different aggregation levels at different viewing distances [35], for example. Signal processing approaches have also been applied to improve the effectiveness of a visualization, for instance, by measuring the difference between the visual salience of a representation and salience of signals in data [37, 46], comparing kernel density estimations between a LOESS curve and different representations [71], and extending a structural similarity index for image compression to data visualization [68]. Signal processing-based approaches have typically been applied to single views, and are generally confined to a predefined set of marks and visual variables (e.g., a line chart, a scatterplot), restricting their applicability for settings like ours.

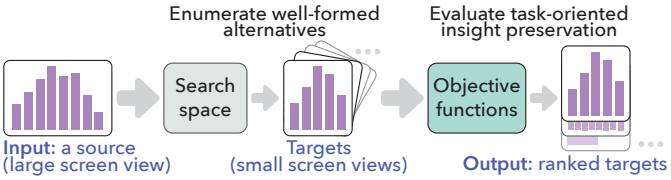


Fig. 2. A pipeline for a responsive visualization recommender

3 Problem Formulation

We propose formulating responsive visualization as a search problem from an input source view to transformed target views, following the characterization proposed by Kim et al. [43]. Consider a recommender that takes a source desktop view as input and returns a ranked set of targets as illustrated in Fig. 2. The first step in creating such a recommender is to define a search space that can enumerate well-formed responsive targets. To generate useful target views from a source (large screen) visualization, a search space should cover common transformation strategies in responsive visualization, such as rescaling, aggregating, binning, and transposing [30, 43].

After enumerating target views, a responsive visualization recommender should evaluate how well each target preserves certain information or "insights." While the term insight can be overloaded [77], a relatively robust way to define insights comes from typologies for describing visualization judgments or patterns [2, 7]. These typologies suggest defining insights around common low-level visual analysis tasks like identifying and comparing data. In an automated design recommendation scenario, these task-oriented insights can be approximated by objective functions (i.e., loss measures) that capture support for common tasks, applied to both the source and target view. Finally, the recommender returns the set of target designs based on how well they minimize these loss measures. We formalize this problem and motivate and define three loss measures that we call *task-oriented insight preservation measures*. In Sect. 5, we describe a prototype visualization recommender in which we implemented the approach.

3.1 Notation

We define a visualization (or a view), V, as a three tuple

$$V = [D_V, C_V, E_V], \tag{1}$$

where D_V is the data used in V, C_V is a visualization specification (defining encodings, chart size, mark type, etc.), and E_V is a set of rendered values that we compute our measures on. For example, suppose a bivariate data set with *GDP* and *GNI* fields (i.e., $D_V = \{x_1, x_2, \ldots, x_n\}$, where $x_i = (x_i.GDP, x_i.GNI)$). C_V maps GDP and GNI to x and y positions of point marks, respectively, producing a scatterplot. The corresponding set of rendered values is a set of Cartesian coordinates on the XY-plane (i.e., $E_V = \{e_1, e_2, \ldots, e_n\}$, where $e_i = (e_i.x, e_i.y)$ is the tuple of rendered values for x_i). Similarly, for a data set containing a field *CO2* (emission) that is mapped to *color*, $e_i.color$ would correspond to the rendered value of $x_i.CO2$. For brevity, we define $D_V.field$

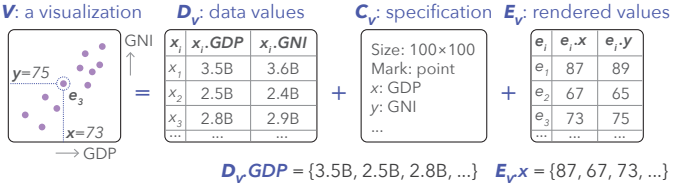


Fig. 3. Our notation for a visualization. Rendered values are defined in the space implied by the visual variable (e.g., pixel space for position or size, color space for color).

as a vector of *field* values and $E_V.channel$ as a vector of rendered values in *channel*. Our notation is also illustrated in Fig. 3

Given a source view $\mathbb{S}$ and a transformation (or target) $\mathbb{T}$, we represent the loss of insight type M from $\mathbb{S}$ to $\mathbb{T}$ as below:

$$Loss(\mathbb{S} \to \mathbb{T}; M) \tag{2}$$

For example, $Loss(\mathbb{S} \to \mathbb{T}; \text{Trend})$ indicates trend loss from $\mathbb{S}$ to $\mathbb{T}$.

4 Task-oriented Insight Preservation Measures

High level criteria for preserving task-oriented insights of a visualization include preserving datum-level information, maintaining comparability of data points, and preserving the aggregate features [7]. We use these distinct classes of information to define task-oriented insight loss measures for approximating how well a responsive transformation preserves support for low-level tasks of identifying data, comparing data, and identifying trend. Our goal is to define a small set of measures that capture important types of low-level tasks a designer might wish to preserve in responsive transformation. Each measure should be distinct (i.e., mostly independent of the others) and should improve accuracy when combined with the others (such as through regression or ML modeling) to predict human judgments about how visualization transformations rank. Together, the measures should outperform reasonable baseline approaches based on simple heuristics. While chosen to cover three important classes of low-level analytic task, the measures we describe are not meant to be exhaustive, as there are many ways one could approximate support for task-oriented insights.

4.1 Identification Loss

Responsive visualization strategies often alter the number of visual attributes of marks that viewers can identify (affecting a low-level identification task [2, 7]). As illustrated in Fig. 4, when the number of bin buckets of a histogram is decreased in a mobile view (a), each bar encodes more information on average than in the desktop view, such that some information about the distribution is lost. Similarly, strategies to adjust graphical density, like aggregating distributions (b) and filtering certain data (c), also reduce the number of identifiable attributes. We use *identification loss* to refer to changes to the identifiability of rendered values between a source view and a target.

Information theory, and in particular Shannon *Entropy* (entropy, hereafter) captures the information in a signal by measuring the minimum number of bits needed to encode it [59]. Given a random variable X, entropy is defined as $H(X) = -\sum_{x \in X} P(x) \log_2 P(x)$. Applying this to visualization, suppose that for a source visualization $\mathbb{S}$, a vector for data field f, $D_{\mathbb{S}}.f = \{x_1.f, \ldots, x_n.f\}$, is mapped to an encoding channel c. The corresponding rendered values $E_{\mathbb{S}}.c = \{e_1.c, \ldots, e_n.c\}$ compose a random variable $U_{\mathbb{S}}.c$ that takes the set of unique values of $E_{\mathbb{S}}.c$ as its outcome space, where the probability of $U_{\mathbb{S}}.c$ taking x is defined as the relative frequency of x in $E_{\mathbb{S}}.c$, formalized as

$$P(U_{\mathbb{S}}.c = x) = Count_i(e_i.c = x)/n \tag{3}$$

$$H(E_{\mathbb{S}}.c) = -\sum_x P(U_{\mathbb{S}}.c = x) \log_2 P(U_{\mathbb{S}}.c = x) \tag{4}$$

We can similarly compute the probabilities of rendered values, $P(U_{\mathbb{T}}.c)$, and the entropy of an encoding channel, $H(E_{\mathbb{T}}.c)$, for a target view $\mathbb{T}$. Finally, we can calculate the identification loss for the channel as the absolute difference in entropy (i.e., $|H(E_{\mathbb{S}}.c) - H(E_{\mathbb{T}}.c)|$), where 0 difference is the identity. The final identification loss from $\mathbb{S}$ to $\mathbb{T}$ is the sum of absolute differences in entropy for each encoding channel c between the two views:

$$Loss(\mathbb{S} \to \mathbb{T}; \text{Identification}) = \sum_c |H(E_{\mathbb{S}}.c) - H(E_{\mathbb{T}}.c)|, \tag{5}$$

4.2 Comparison Loss

Responsive transformations like resizing or scaling a view or aggregating data can alter the number of possible data comparisons that a user can make and how perceptually difficult they are (affecting a low-level comparison task [2, 7]). For instance, in Fig. 5, resizing (a) diminishes the magnitude of difference between two highlighted data points in the small screen design. In a mobile design with aggregation (b), viewers are no longer able to make each comparison that is available in the large screen view. This motivates estimating how similarly viewers are able to discriminate between pairs of points in a target view compared to the source view, which we refer to as *comparison loss*.

Empirical visualization studies (e.g., [44, 64]) often operationalizes accuracy as the viewer's ability to perceive relationships between pairs of values. While simpler scalar statistics like a sum or mean might suffice under some transformations, a method that preserves the distribution of distances will be more robust to transformations that change the number of data points or scales (e.g., log-scale). We operationalize comparison loss as the difference in pairwise discriminability, measured using Earth Mover's Distance (EMD), between the source and a target in each encoding channel used in a visualization:

$$Loss(\mathbb{S} \to \mathbb{T}; \text{Comparison}) = \sum_c EMD(B_{\mathbb{S}}.c, B_{\mathbb{T}}.c'), \tag{6}$$

where $B_{\mathbb{S}}.c$ and $B_{\mathbb{S}}.c'$ are the *discriminability distributions* of the source and target views in encoding channel c and c', respectively, that encode the same data field.

Given a source visualization $\mathbb{S}$, we define the discriminability distribution $B_{\mathbb{S}}.c$, of an encoding channel c for a view $\mathbb{S}$, as the set of distances between each pair of rendered values ($E_{\mathbb{S}}.c$) of $\mathbb{S}$ in terms of c. This is formalized as

$$B_{\mathbb{S}}.c = \{d_c(e_i.c, e_j.c) : e_i.c, e_j.c \in E_{\mathbb{S}}.c\}, \tag{7}$$

where $d_c(\cdot,\cdot)$ is a distance metric for the encoding channel c.

Distance metrics: Ideally, comparison loss should account for differences in how well visual channels support perception of numerical values. Informed by visual perception models, we select several distance metrics intended to provide a rough proxy of the perceptual difference between two visual signals. While visual variables can have interaction effects [9, 60, 64], for simplicity in demonstrating our approach, we limit our use of perceptual distance metrics to encoding

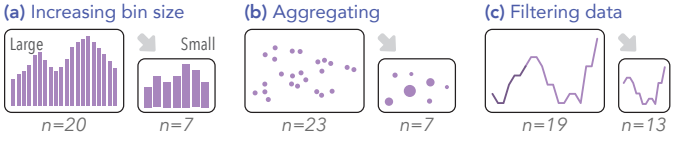


Fig. 4. Responsive transformations that may cause identification loss.

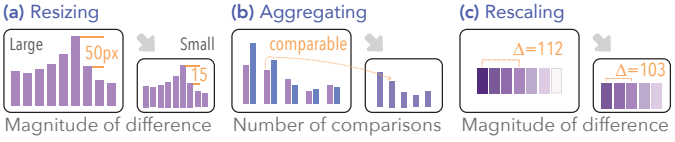


Fig. 5. Responsive transformations that may cause comparison loss.

channel specific measures. However, as the state-of-the-art in predicting effects of visual variable interactions develops, our approach could be amended to consider combinations.

For position channels, we use the absolute difference between two position values (in pixel space), as human vision is highly accurate in discriminating positions according to Stevens' power law [62, 63] and empirical studies [27, 28]:

$$d_{position}(e_i.position, e_j.position) = |e_i.position - e_j.position| \quad (8)$$

We measure distance in a size channel using the absolute difference between two size values (in pixel) raised to the estimated Stevens' exponent of 0.7 [62, 63]:

$$d_{size}(e_i.size, e_j.size) = |e_i.size - e_j.size|^{0.7} \quad (9)$$

We calculate the Euclidean distance in the perceptual color space CIELAB [20] (CIELAB 2002):

$$d_{color}(e_i.color, e_j.color) = \sqrt{(e_i.L - e_j.L)^2 + (e_i.a - e_j.a)^2 + (e_i.b - e_j.b)^2}, \quad (10)$$

where L, a, and b represent $L*$, $a*$, and $b*$ in CIELAB space.

Lastly, for shape encodings, we employ a perceptual kernel [16], a (symmetric) matrix of pairwise distances between visual attributes. The i,j-th element in the perceptual kernel for shape is the empirical probability of discriminating shape i from shape j based on an online crowdsourced experiment in which workers completed a triplet discrimination task where they chose the most dissimilar shape out of three shapes. Formally, our shape distance metric can be stated as:

$$d_{shape}(e_i.shape, e_j.shape) = P(e_i.shape \text{ is discriminated from } e_j.shape) \quad (11)$$

Comparing discriminability distributions: To quantify the discrepancy between the discriminability distributions of encoding channel c and c' (mapping the same field) for the source $\mathbb{S}$ and target $\mathbb{T}$ (i.e., $B_{\mathbb{S}}.c$ and $B_{\mathbb{T}}.c'$, respectively), we compute Earth Mover's Distance [69] (EMD or Wasserstein distance). We use EMD, which measures the minimum cost to transform a distribution to another distribution, because it is non-parametric, symmetric, and unbounded. An EMD of 0 is the identity, and the greater the EMD is, the more different the two distributions are. Thus, the comparison loss between the source view $\mathbb{S}$ and a target view $\mathbb{T}$ is the sum of the EMD between their discriminability distributions in each encoding channel, formalized in Equation 6.

4.3 Trend Loss

Responsive transformations like disproportionate rescaling and changes to binning may impact the implied relationship (or trend) between two or more variables represented in a target view compared to the source view (affecting low-level trend identification [7]). As shown in Fig. 6, different aspect ratios can alter the magnitude of the slope of a trend, and modifying bin size affect the amount of distributional information available. We use *trend loss* to refer to changes in the implied trend from the source to a target.

To capture representative data patterns while avoiding influences of noise, our trend loss first estimates trend models between the source and target views using LOESS. We then compare the area (or volume) of the estimated trends because it is more sensitive to details that simpler methods (e.g., the difference between regression coefficients) might ignore. We define trend models for the quantitative encoding channels in our scope (position, color and size):

- $e_y \sim e_x$: a 2D trend of y on x as appears in a simple scatterplot, line chart, or bar graph.
- $e_{color} \sim e_x + e_y$: a 3D trend of color on x and y like a heatmap or a scatterplot with a continuous color channel
- $e_{size} \sim e_x + e_y$: a 3D trend of size on x and y (e.g., a scatterplot with a continuous size encoding)

After calculating trend models for a source and target, we can define trend loss as the sum of the relative area between curves (or volume between surfaces) of the estimated trends in each trend model (m). This is formalized as:

$$Loss(\mathbb{S} \to \mathbb{T}; \text{Trend}) = \sum_m A(LOESS(m_{\mathbb{S}}), LOESS(m_{\mathbb{T}})) \quad (12)$$

where A stands for the relative area between curves (ABC) between the source and target trends ($m_{\mathbb{S}}$ and $m_{\mathbb{T}}$), normalized by dividing by the area under the curve of the source trend for a 2D model. For a 3D model, A is the relative volume between surfaces (VBS), which is the VBS of the source and target trends divided by the volume under the surface of the source trend.

We estimate the trend models using LOESS regression [12] as it is non-parametric. We use uniform weights and bandwidth of 0.5 [12]. LOESS regression returns an estimate at each observed value of the independent variable(s) (as an array of coordinates): an estimated curve for a 2D model and an estimated surface for a 3D model. Thus, when source and target views have different chart sizes or different sets of rendered values for the independent variable(s), it is difficult to directly compare the LOESS estimations. As shown in Fig. 7a, we

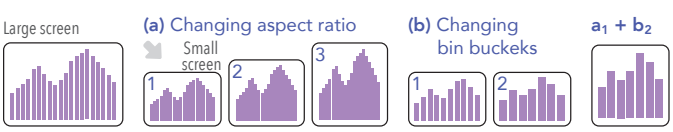


Fig. 6. Responsive transformations motivating trend loss.

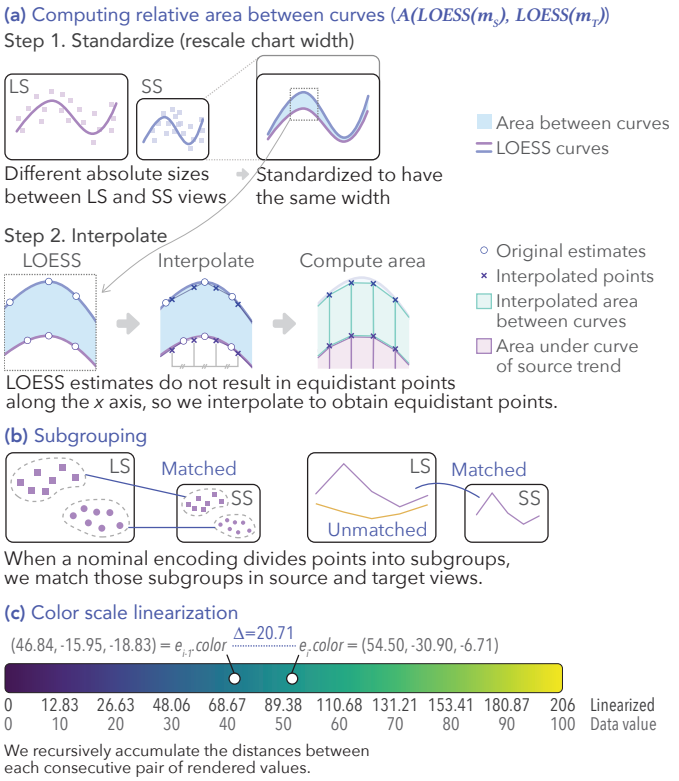


Fig. 7. Components of computing trend loss. (a) Calculating area between curves by standardizing chart size and interpolating break points. (b) Dividing and matching subgroups. (c) Linearizing color scale. LS is large screen, and SS is small screen.

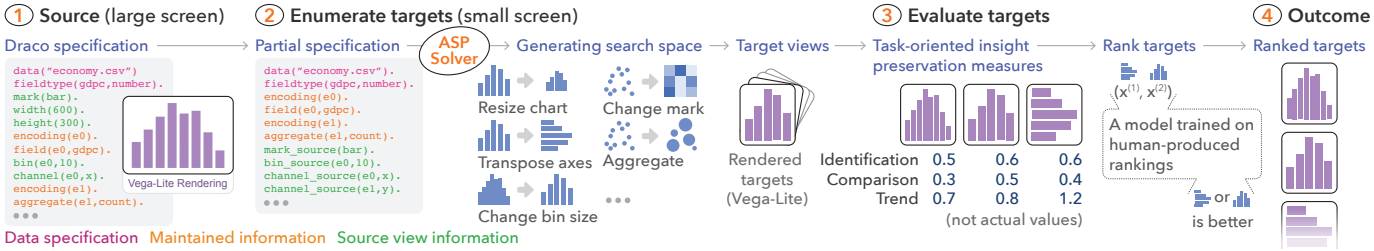


Fig. 8. Prototype pipeline. (1) The full specification of an input source view in ASP. (2) Enumerating targets by extracting a partial specification of the source view and generating a search space using an ASP solver. (3) Evaluating targets by computing our loss measures and ranking them using a model trained on human-produced rankings. (4) Ranked targets.

first standardize the chart sizes of two views by rescaling an estimated LOESS curve or surface in a target view to have the same chart width with the source. Then, we interpolate the LOESS curve to have equal distances between two consecutive coordinates for a 2D model (Fig. 7b). We interpolate on 300 breakpoints in a 2D model by default, where one breakpoint corresponds to one to three pixels in many Web-based visualizations. For a 3D model, we interpolate 300×300 breakpoints from a LOESS surface in a similar way. Given these interpolations for the LOESS curves (or surfaces) in the source and target, we obtain the ABC (or VBS) segment at each breakpoint.

Subgroups: When a nominal variable encoded by color or shape divides the data set into subgroups, viewers might naturally consider each subgroup's trend independently. To distinguish trends implied by subgroups, we first identify and match subgroups which occur in both the source and target views by looking at their nominal data values, as depicted in Fig. 7b. Then, we compute the relative ABC (or VBS) of each subgroup and combine them by taking their average.

Color scale linearization: Although a continuous color scale encodes a unidimensional vector, color is often modeled on a multi-dimensional space (e.g., RGB, CIELAB), which makes it complex to estimate a LOESS surface. Similar to how common color schemes such as *viridis* or *magma* are designed to be perceptually uniform by keeping equidistance in a perceptual color space between two consecutive color points [67], we can make use of the Euclidean distance between rendered color values in CIELAB to linearize a 3D color scheme. Specifically, we recursively accumulate the distances between each consecutive pair of rendered values to create a unidimensional vector. In Fig. 7c, we show how the linear value of i-th color point is computed from that of $i-1$-th point; we take the calculated value of the $i-1$-th point and add to it the distance between the $i-1$-th and i-th points. The first color point is assigned as zero.

5 Prototype Responsive Visualization Recommender

To implement our task-oriented insight preservation measures, we developed a prototype responsive visualization recommender that enumerates and evaluates responsive designs (or targets). As shown in Fig. 8, given an input source (large screen) view, our recommender first converts it to a partial specification, and then generates a search space of small screen targets based on the partial specification. We adopt the desktop-first approach that visualization authors have described using [30, 43]. Finally, the recommender computes our measures between the source view and each target to rank those targets using an ML model trained on human-labeled rankings.

5.1 Enumerating Target Views

To enumerate target views, we need a formal grammar for representing visualization specifications and formulating a search space. We use Answer Set Programming (ASP) [8], particularly by modifying Draco [50]. ASP is a declarative programming language for complex search problems (e.g., satisfiability problems) that encodes knowledge as facts, rules, and constraints. Rules generate further facts, and constraints prevent certain combinations of facts. Formalized in ASP, for example, Draco has a rule that if an encoding is binned, then it is discrete, and a constraint that disallows logarithmic scale on a discrete encoding [50]. A constraint solver then solves an ASP program (the partial specification of a source view and our search space), returning stable sets of non-conflicting facts (enumerated target views with different transformation strategies). We use Clingo [21, 22] as our solver.

Converting to a partial specification: Our recommender converts the full specification of an input source view to a partial specification to allow applying responsive transformation strategies. We maintain the data specification (data file, data field definitions, and the number of rows) and encoding information (e.g., count aggregation, association of data field) that are not changed under transformation. We indicate the rest of the specification (mark type, chart size, and encoding channels) as information about the source view to constrain responsive transformation strategies (e.g., constraining possible mark type replacement, allowing for swapping position encodings for axis-transpose).

Generating a search space: Our goal in generating a search space is to produce a set of reasonable targets that a responsive visualization author might consider given a source view. We generate a search space by automatically applying responsive visualization transformations recently observed in an empirical study of common responsive visualization design strategies [43] to a source visualization. Our prototype implements rescaling, aggregation, binning, transposing, and select changes to marks and encodings. For rescaling, we fix the width of target views and vary heights, in the range from the height resulting from proportionate rescaling to the height that forms the inverse aspect ratio with an increment of 50 px. For example, if the source view has a width of 600 px and a height of 300 px (an aspect ratio of 2:1) and the width of target views is fixed at 300 px, then the height varies from 150 px (2:1) to 600 (1:2) by 50 px (i.e., 150, 200, ..., 550, 600 px). Given a disaggregated source view, we generate alternatives by applying binning (max bin buckets of 25, 15, and 5) and aggregation (count, mean, median, sum) as graphical density adjustment strategies. We also generate alternatives by transposing axes (i.e., swapping x and y position channels). Finally, in line with the observation of prior work that responsive visualization authors occasionally substituted mark types when adding an encoding channel for aggregation, we allow a mark type change in scatterplots from a point mark to a rectangle (heatmap). We formulate these strategies in ASP format and add them to Draco [50].

5.2 Evaluating and Ranking Targets

To evaluate enumerated targets, we calculate our loss measures on rendered values after rendering source and target views using Vega-Lite [56]. Then, we obtain rendered values, E_V, of a visualization V by gleaning Vega [57] states (a set of raw rendered values [66]). We implemented the loss measures in Python using SciPy [70]'s `stats.entropy` and `stats.wasserstein_distance` methods for entropy and EMD, respectively. To compute LOESS regression, we use the `LOESS` package [11]. Finally, to rank the enumerated targets, we combine the computed loss values by training ML models, which we detail in Sect. 6. We use ML models for ranking instead of formalizing them in ASP because our measures are not declarative (not rule-based).

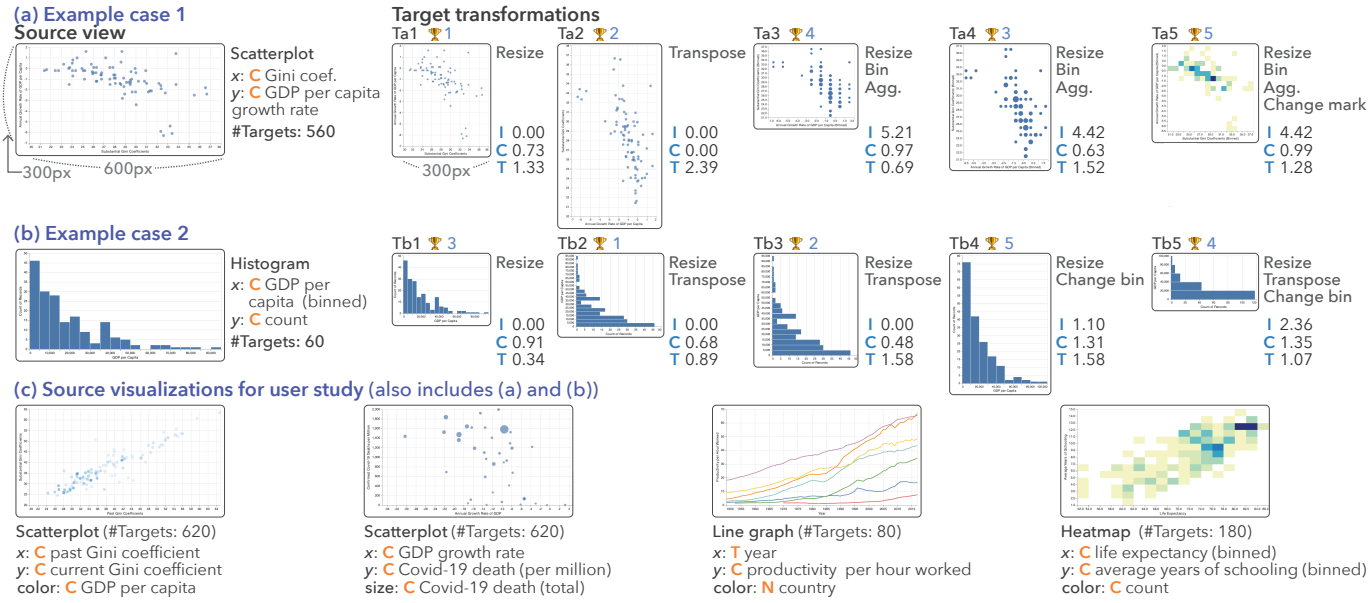


Fig. 9. (a, b) Example target transformations enumerated by our prototype responsive visualization recommender (total size of search space per source given as #Targets). 🏆 indicates rankings of each five targets per source view predicted by our best model (see Sect. 6.3). (c) Source visualizations for our user study (also includes a and b). Sources views have width of 600px and height of 300px. The width of targets is fixed as 300px. Data sets are from *Our World in Data* [26, 53, 54]. Continuous, Nominal, Temporal, Identification loss, Comparison loss, and Trend loss.

5.3 Examples

We introduce two example cases of transformations generated by our prototype and describe how our measures distinguish target views.

5.3.1 Case 1: Simple scatterplot

In the source scatterplot (Fig. 9a), each point mark represents a country, and x and y positions encode Gini coefficients and annual growth rate of GDP per capita of different countries, respectively. The first example transformation (Ta1) is simple resizing. The second target view (Ta2) is transposed from the source view while keeping the size. The third and fourth target views (Ta3 and Ta4) are resized, binned in x and y scales, and aggregated by count, so the size of each dot represents the number of data points in the corresponding bin bucket. In the fifth target (Ta5), the mark type is changed from point to rectangle in addition to resizing, binning, and aggregating, and the color of each rectangle encodes the number of data points in that cell.

Because Ta1 and Ta2 perfectly preserve the number of identifiable rendered values, identification loss is zero. Ta4 and Ta5 have more identifiable points than Ta3 (due to their smaller bin size), so they have smaller identification loss. While Ta1 has disaggregated values, Ta4 better preserves the distances between points in terms of position encoding, so it has smaller comparison loss. Compared to the source view, the implied trend given x and y positions in Ta1 has a more similar slope and hence smaller trend loss than Ta2, whereas Ta2 preserves the differences in the position encodings, resulting in zero comparison loss. Similarly, Ta3 has a smaller trend loss than Ta4 because Ta3 better preserves the visual shape of the distribution in the source view.

5.3.2 Case 2: Histogram

The source histogram in Fig. 9b shows the distribution of GDP per capita of different countries. There are 23 bins along the x axis and each bar height (y position) represents the number of countries in the corresponding bin. The first target view (Tb1) is resized. The second and third target views (Tb2 and Tb3) are transposed with different resizing. In the fourth and fifth target views (Tb4 and Tb5) bin sizes are changed from (23 to 10 and 5, respectively), with Tb5 transposed.

As Tb1, Tb2, and Tb3 have no changes in binning, they have zero identification loss, whereas Tb4 and Tb5 has greater identification loss proportional to their bin sizes. While Tb1, Tb2, and Tb3 have the same binning, Tb3 has the most similar differences between bar heights and bar intervals in pixel space, so it has the smallest comparison loss among them. Transposing axes (Tb3) better preserves the resolution for comparison (i.e., chart height and width), often resulting in the smaller comparison loss than other similarly transformed targets. Tb5 has smaller trend loss than Tb4 as it shows a similar aspect ratio to the source view, though inverted, as implied by x and y positions.

6 Model Training and Evaluation

A responsive visualization recommender should combine loss measures to rank a set of targets by how well they preserve task-oriented insights. For our prototype recommender, we train machine learning models to efficiently combine our loss measures and rank enumerated targets. We describe training data collection, model specification, and results.

6.1 Labeling

We obtained training and test data consisting of ranked target views for a set of source views using a Web-based task completed by nine visualization experts. As shown in Fig. 10a, each labeler was assigned one out of three trial sets and performed 36 trials, with each trial asking them to rank five target transformations (small screen) given a source visualization (large screen).

Task materials: To create instances for labeling, we selected six desktop visualizations (source views) as shown in Fig. 9c. Our goal was to include different chart types, multiple encoding channels for identification and comparison losses, and different types of examples for trend loss (e.g., 2D/3D models, subgroups, color scale linearization). Our prototype generates 60 to 620 target transformations (2,120 in total) for these six source views. We generated three sets of 30 target views per source view for labeling, using quintile sampling per preservation (loss) measure, to ensure relatively diverse sets of targets. After sorting targets in terms of each of our three measures, we sampled two targets from each quintile of the top 100 targets per measure, as depicted in Fig. 10c. We took the top 100 targets after inspecting the best ranked views per measure for each source view, to avoid labeling examples that might be obviously inferior. Because identification loss is measured using entropy and is primarily affected by how data are binned, certain source views had fewer than five unique discrete values within top 100 targets. In this case, we proportionately sampled each discrete value.

After sampling 30 targets for a source view in a trial set, we randomly divided them into six trials (but fixed these trials between labeler in the

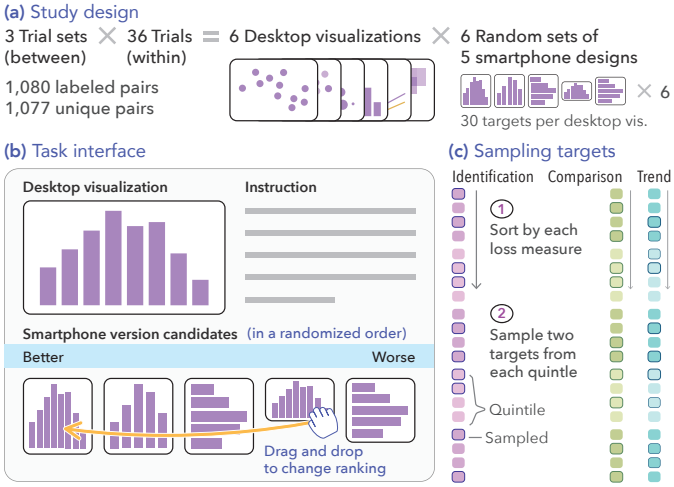


Fig. 10. (a) Study design. (b) Task interface. (c) Quintile sampling of targets for task materials.

Features		Chart types			
Aggregated	**D**isaggregated				
Insight type	Enc./Model	Scatterplot	Bar graph	Line chart	Heatmap
Identification	*x*	Required			
	y				
	size	Optional			
	color				
	shape				
Comparison	*x*				
	y				
	size				
	color				
	shape				
Trend	*y*~*x*				
	size~*x*+*y*				
	color~*x*+*y*				

Table 1. The set of features for our ML models by each chart type. These features are either concatenated or differentiated for each pair of targets. **A**ggregated features are the sum of the corresponding **D**isaggregated features. Pink, bold-bordered circles represent required features, and yellow, light-bordered circles optional encoding-specific features.

same trial set), so we had 1,080 pairs (1,077 unique pairs[1]) labeled by three people each. We randomly assigned each trial set to labeler (3 trial sets (between) × 6 source views × 30 targets (within), Fig. 10a).

Labelers: All five authors, who have considerable background in visualization design and evaluation, and an additional convenience sample of four visualization experts (representing postdoctoral researchers and graduate students in visualization) participated in labeling. All labelers worked independently.

Labeling task: Each labeler was asked to imagine that they were a visualization designer for a responsive visualization project, tasked with ranking a set of small screen design alternatives created by transforming the source. Their goal was to consider what would be an appropriate small screen design that would also preserve insights or takeaways conveyed in the desktop version as much as possible.

The study interface is shown in Fig. 10b. Each labeler completed 36 trials (6 desktop visualizations × 6 sets of 5 smartphone design candidates). In each trial, the desktop visualization and five smartphone design candidates were shown, and labeler ranked the candidates by dragging and dropping them into an order. Trial order was randomized.

Aggregating labels: From the task, we collected human-judged rankings of 1,080 pairs each of which was ranked by three labelers. To produce our training data set, we aggregated the three labels obtained from the three labelers of each pair into a single label representing the majority opinion, such that that for the i-th pair $\mathbf{x}_i = (\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)})$, the label y_i is 1 if $\mathbf{x}_i^{(1)}$ is more likely to appear higher than $\mathbf{x}_i^{(2)}$, and -1 otherwise.

$$y_i = \begin{cases} 1, & \text{if } \mathbf{x}_i^{(1)} \text{ more often appears higher than } \mathbf{x}_i^{(2)} \\ -1, & \text{otherwise} \end{cases} \tag{13}$$

To avoid a biased distribution of training data as well as minimize the ordering effect within each pair, we randomized the order of pairs so that half of the pairs are labeled as 1 and the other half as -1, which naturally sets the baseline training accuracy of 50%.

6.2 Model Description

Prior approaches to visualization ranking problems (e.g., Draco-Learn [50], DeepEye [48]) utilize ML methods that convert the ranking problem to a pairwise ordering problem, such as RankSVM (Support Vector Machine) [29] and the learning-to-rank model [10]; we adopt a similar approach. A model, f, takes as input a pair of objects, $\mathbf{x} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)})$, and returns their orders (i.e., either $\mathbf{x}^{(1)}$ or $\mathbf{x}^{(2)}$ ranks higher).

$$f(g(\mathbf{x}^{(1)}, \mathbf{x}^{(2)})) = \begin{cases} 1, & \text{if } \mathbf{x}^{(1)} \text{ appears higher in the ranking} \\ -1, & \text{if } \mathbf{x}^{(2)} \text{ appears higher in the ranking} \end{cases}, \tag{14}$$

where $g(\cdot,\cdot)$ is a mapping function that combines the features from a pair of objects. We consider vector difference and concatenation for g. Our models take two target views representing transformations of the same source view and return the one with higher predicted ranking, as depicted in Fig. 8.3.

Features: We define the feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ where each row corresponds to a pair of target visualizations and columns represent the features (converted by g). We use our proposed loss measures as the features (Table 1). Aggregated features (**A**) refer to our three loss measures: identification, comparison, and trend loss, as described in Equations 5, 6, and 12 (Sect. 4). Disaggregated features (**D**) refer to the components of the aggregated features (e.g., the EMD value in each encoding channels for comparison loss). We standardized all features.

Model training: We train SVM with a linear kernel, K-nearest neighborhood (KNN) with $k = 1, 10$, logistic regression, decision tree (DT), and a Multilayer Perceptron (MLP) with four layers and 128 perceptrons per layer, similar to other recent applications of ML in data visualization (e.g., Hu et al. [32], Luo et al. [48]). We also train ensemble models of DTs: random forest (RF) with 50, and 100 estimators, Adaptive Boosting (AB), and gradient boosting (GB). Given the moderate number of observations (1,067) in our data set, we use leave-one-out (LOO) as a cross validation iterator to obtain robust training results. We used Scikit-Learn [51] for training.

Baselines: In addition to the natural baseline of 50% (random), we include two simple heuristic-based baselines to evaluate the performance of our models. The first baseline (**B**1) includes the changes in chart width and height between a target and its source, capturing an intuition about maintaining size and aspect ratio. The second baseline (**B**2) is whether x and y axes are transposed, capturing an intuition that, of the strategies in our search space, transposing is the most drastic change.

6.3 Results

All the experimental materials, and files used for analysis are included in the supplementary materials, available at `https://osf.io/jcvbx`.

6.3.1 Rank correlation between loss measures

To ensure that our loss measures capture different information about transformations, we compute and inspect rank correlations between each pair of aggregated, and each pair of disaggregated measures. If two different loss measures produce highly similar rankings of target views, then one of them might be redundant. Our measures tend to be orthogonal to each other (see Fig. 11), with Kendall rank correlation

[1]Each source view had 180 pairs, but the histogram source view with 60 transformations has 177 unique pairs.

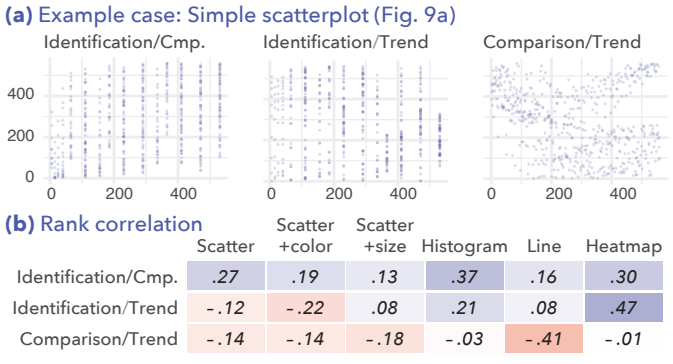
(a) Prediction accuracy

Models	Features	Disaggregated		Aggregated		Disagg. + Agg.		B1 (chart size change)		B2 (axes-transpose)	
	Mappings	concatenate	difference	concatenate	difference	concatenate	difference	concatenate	difference	concatenate	difference
1-Nearest Neighborhood		78.73	77.88	71.60	67.48	77.79	77.13	51.08	49.02	62.32	59.70
K-Nearest Neighborhood		76.48	77.79	72.73	72.26	76.01	77.32	55.11	50.42	62.61	62.32
Logistic Regression		78.63	78.07	75.91	**76.38**	78.63	77.98	53.70	54.08	62.42	62.42
SVM Linear		78.35	78.82	75.35	76.01	78.44	78.35	55.01	59.04	62.89	63.17
Decision Tree		76.19	75.91	70.20	67.95	76.19	74.70	50.05	51.08	63.17	63.17
Rrandom Forest (e=50)		83.04	**82.38**	76.66	74.70	82.19	**82.57**	50.89	50.61	61.20	62.89
Random Forest (e=100)		**84.07**	82.29	**77.41**	74.32	82.76	82.38	51.55	50.98	62.04	63.07
Adaptive Boosting		81.16	79.01	72.73	73.01	79.94	78.73	56.42	53.05	62.42	63.17
Gradient Boosting		81.82	80.88	77.32	75.26	**82.94**	80.04	53.51	53.05	63.17	63.17
Multilayer Perceptron		81.72	82.10	77.23	76.29	80.97	81.07	54.92	53.61	61.95	62.04

(b) Feature importances

Features		Importance*
Ident.	*y*	.159
	x	.153
	color	.067
	size	.026
Cmp.	*y*	.141
	x	.114
	color	.092
	size	.038
Trend	*y~x*	.136
	color~*x*+*y*	.055
	size~*x*+*y*	.020

Table 2. (a) Prediction accuracy of our models, averaged over LOO cross validation. Other performance measures (AUC score and F1-score) appeared similarly to accuracy. (b) Average importance of **D**isaggregated features (g = difference) measured by impurity-based importance from training a random forest model (e = 50) 10 times.



(b) Rank correlation

	Scatter	Scatter +color	Scatter +size	Histogram	Line	Heatmap
Identification/Cmp.	.27	.19	.13	.37	.16	.30
Identification/Trend	-.12	-.22	.08	.21	.08	.47
Comparison/Trend	-.14	-.14	-.18	-.03	-.41	-.01

Fig. 11. (a) Joint distributions of rankings of target views in each pair of aggregated features (the source visualization is shown in Fig. 9a). (b) Kendall rank correlation coefficients for targets of our source views in Fig. 9. Cmp (comparison).

coefficients [42] between -0.41 and 0.47. The same pattern is observed for the disaggregated measures with overall correlation coefficients mostly between -0.5 and 0.5 (see supplementary material).

When the chart type of a source view allows a few, limited responsive transformation strategies due to its own design constraints (e.g., line chart, heatmap), the correlation between measures appear slightly higher than the other chart types. For example, it is often impossible to add a new encoding channel through aggregation or binning in a line chart. This makes the line chart more sensitive to chart size changes, resulting in relatively higher negative rank correlation between comparison and trend loss. Similarly, different binning levels in a heatmap can affect both one's ability to identify data points in different encoding channels and to recognize a trend implied by x and y on color channels (i.e., e_{color} $e_x + e_y$), leading to a slightly higher positive rank correlation between identification and trend loss.

6.3.2 Monotonicity

Ranking problems through pairwise comparison assume that the partial rankings used as input are consistent with the full ranking (monotonicity of rankings) [29, 36]. In other words, we need to ensure that the partial pairwise rankings that we calculate based on aggregated expert labels can yield a monotonic full ranking. Comparison sorting algorithms can be used to determine whether a monotonic full ranking can be obtained from pairwise rankings, as a comparison sort will only result in a monotonic ranking if the principle of transitivity ($a > b \wedge b > c \implies a > c$) and connexity ($\forall a$ and $b, a \leq b \vee b \leq a$) hold.

To confirm whether our expert labels satisfy the monotonicity assumption, we first sort the five target views in each of our 108 trials, using the ten aggregated pairwise rankings as a comparison function. Next, we check whether each consecutive pair in the reproduced ordering conflicts with the aggregated expert labels, because if a pair in the reproduced ordering is not aligned with the aggregated label, that trial violates the monotonicity assumption. 102 out of 108 trials (94.44%) in our data set had fully monotonic orderings. Of the six orderings which are not fully monotonic, five are partially monotonic with only one misaligned pair each (out of the ten ordered pairs). The other non-monotonic ordering (a trial with line chart as the source view) had multiple conflicts; we dropped this ordering from our training data, resulting in 1,070 training pairs (1,067 unique training pairs).

6.3.3 Training results

Model performance: Overall, our models with disaggregated (**D**) and aggregated features (**A**) achieved prediction accuracy greater than 75% (Table 2a), showing the utility of our measures in ranking responsive design transformations. Ensemble models (RF, AB, and GB) with **D** features resulted in the highest overall accuracy (above 81%) because they iterate over multiple different models and we have a relatively small number of features. In particular, RF with **D** and 100 estimators showed highest accuracy of 84%. Our neural network model (MLP) also provided comparable performance to the ensemble models.

Models with **D** features in general obtained higher accuracy than **A** features, and combining them (**D** + **A**) did not provide significant gain in accuracy. Although they had only three features, our models with **A** features showed reasonable accuracy of up to 77.4% (g = concatenate) and 76.4% (g = difference). For mapping functions, concatenation performed slightly better than difference for our best performing models (RF).

Our models all outperformed those with both baselines features (**B**1 and **B**2), indicating that our loss measures capture information that simple heuristics, such as changes in chart size or axes transposition, are unable to capture. When we trained the best performing model (RF) with the features of only a single loss criterion (e.g., only trend), accuracy ranged from 52.6% to 79.7%, implying that our measures are more useful when combined than when used individually. As hypothetical upper bounds for accuracy, training and testing the model on the same data set resulted in accuracy from 84% (KNN) to 100% (RF).

Feature importance: To understand how the different loss measures function in our models, we inspected the importance of each disaggregated feature (mapping function g = difference) using the impurity-based importance measure (average information gain) by training a random forest model with 50 estimators (average over 10 training iterations). As shown in Table 2b, features related to position encodings (x, y, $e_x \sim e_y$) in general seem to have higher importance, which makes sense given their ubiquity in our sample.

Predicted rankings of example cases: Using the best prediction model (RF with 100 estimators), we predicted the rankings of example cases described in Sect. 5.3. Transformations from the simple scatterplot example (Fig. 9a) are ranked as: Ta1 (resizing), Ta2 (transposing axes), Ta4, Ta3 (binning, resizing, aggregation), and Ta5 (binning, resizing, aggregation, mark type change). Ta1 appears higher in ranking than Ta2 because Ta2 has higher trend loss, while Ta1 slightly sacrifices comparison loss. Ta4 is ranked in a higher position than Ta5 because Ta5 has higher comparison and trend loss. Responsive transformations

from the histogram example (Fig. 9b) are ranked as: Tb2 (transposing axes, resizing), Tb3 (transposing axes, resizing), Tb1 (resizing), Tb5 (resizing, changing bin size), and Tb4 (resizing, changing bin size). The transposed views (Tb2 and Tb3) are ranked higher than Tb1 probably because the model has more emphasis on comparison loss as the feature importance (Table 2b) shows. The ordering between Tb5 and Tb4 can be backed by the smaller trend loss of Tb5 while the difference in comparison loss between them appears subtle.

7 Discussion and Future Work

7.1 Extending and Validating Our Preservation Measures

We devised a small set of measures for three common low-level tasks in visualization use and found that they can be used to build reasonably well-performing ML models for ranking small screen design alternatives given a large screen source view. Our measures are not strongly correlated, and removing some of the measures results in lower predictive accuracy. However, there are other forms of prominent task-oriented insights that could extend our approach if approximated well, such as clustering data points or identifying outliers. As our measures lose information by processing rendered values, future work could estimate task-oriented insights with different methods, such as extracting and directly comparing image features from rendered visualizations.

There are also opportunities to strengthen and extend our measures through human subject studies. These include more formative research with mixed methods to understand heuristics and other strategies that visualization authors and users employ to reason about how well a design transformation preserves important takeaways. In addition, future work could conduct perceptual experiments that more precisely estimate human baselines for identification, comparison, and trend losses. We also used simple approximations of perceptual differences in position, size, and color channels which could be improved through new experiments specifically designed to understand how perception is affected on smaller screen sizes, adding to work like examining task performance on smaller screens by different chart types [6] and comparing task performance between small and large screens [3, 4]. A limitation is that our experiment was conducted on desktop devices. Future work could test on mobile devices, as well as explore mobile-first design contexts, as our measures are designed to be symmetric.

7.2 Responsive Visualization Authoring Tools

Our work demonstrates how task-oriented insight preservation can be used to rank design alternatives in responsive visualization. To do so, we formulated and evaluated our insight preservation measures on a search space representing common responsive visualization design strategies and mark-encoding combinations. However, our work should be extended in several important ways to support a responsive visualization authoring use case.

First, while more drastic encoding changes than those supported by our generator are rare in practice [43], this might be because responsive visualization authoring is currently a tedious process and authors satisfice by exploring smaller regions of the design space. There are many strategies that could be added to a search space like the one we defined, and used to evaluate our measures as well as to learn more about how authors react when confronted with more diverse sets of design alternatives. For example, while we mainly consider single-view, static visualizations, many communicative visualizations employ multiple views and interactivity [33, 34, 58]. Ideally a responsive visualization recommender should be able to formulate related strategies (e.g., rearranging the layout of multiple views, omitting an interaction feature, editing non-data ink like legends). Recommenders may need to consider further conditions such as consistency constraints for multiple views [52], effectiveness of visualization sequence [45], semantics of composite visualizations [38], and effectiveness of interactive graphical encoding [55]. As indicated in Kim et al. [43], loss measures should be able to address concurrency of information because rearranging multiples views (e.g., serializing) can make it difficult to recognize data points at the same time on small screen devices. In addition, they should also account for loss of information that can only be obtained via user interaction (e.g., trend implied by filtered marks).

We envision our measures, and similar measures motivated to capture other task-oriented insights, being surfaced for an author to specify preferences on in a semi-automated responsive visualization design context. Because what our measures capture is relatively interpretable, authors may find it useful to customize them for certain design tasks, such as prioritizing one measure or changing how information is combined to capture identification, comparison, or trend loss. This is a strength of our approach relative to using a more "black-box" approach where model predictions might be difficult to explain.

7.2.1 Extending ML-based approaches

The human labelers in our experiment, including the authors, seemed to at times use strategies or heuristics such as preferring non-transposed views in their rankings or trying to minimize changes to aspect ratio for some chart types. However, models with our loss measures as features perform better than heuristic approaches like detecting axes-transpose and chart size changes, implying that task-oriented insights may be the right level at which to model rankings. As an extension, future work might learn pre-defined costs for different transformation strategies to reduce the time complexity of evaluating task-oriented insights preservation, similar to the approach adopted by Draco-Learn [50] which obtained costs for constraint violation. Learning such pre-defined costs may also enable better understanding how each responsive design strategy contributes to changes in task-oriented insights. An alternative approach could be to use our loss measures as cost functions and optimize different strategies to reduce them as MobileVisFixer [74] fixes a non-mobile-friendly visualizations for a mobile screen by minimizing heuristic-based costs. As recent deep learning models [49, 75] have performed well in visualization ranking problems, future work may further elaborate on those models. In doing so, one could combine our measures with image features (e.g., ScatterNet [49]) or chart parameters (e.g., aspect ratio, orientation [75]).

As noted in Sect. 6.3.2, there were a few partial and not fully monotonic orderings in our data set. A better model might ignore this assumption and try to identify highly recommendable transformations or classify them into multiple ordinal classes, yet this might come up with lower interpretability about recommendations due to a lack of explicit ordinal relationship between transformations.

7.3 Generalizing Our Measures to Other Design Domains

Our approach to task-oriented insight preservation is likely to be useful in visualization design domains beyond responsive visualization, like style transfer and visualization simplification, although other domains may also require different transformation strategies. Style transfer, for instance, may involve techniques like aggregation but is more likely to change visual attributes of marks or references. While our loss measures are designed to be low-level enough to apply relatively generically to visualizations, their precise formulation and the combination strategy might warrant changes in other domains. For example, in visualization simplification, minimizing trend loss is likely to be more important than preserving identification and comparison of individual data points. Style transfer often focuses on altering color schemes, size scales, or mark types [24] which can result in different discriminability distributions, so it might put more emphasis on comparison loss.

8 Conclusion

Responsive visualization transformations often alter task-oriented insights obtainable from a transformed view relative to a source view. To enable automated recommenders for iterative responsive visualization design, we suggest loss measures for identification, comparison, and trend insights. We developed a prototype responsive visualization recommender that enumerates transformations and evaluates them using our measures. To evaluate the utility of our measures, we trained ML models on human-produced orderings that we collected, achieving accuracy of up to 84.1% with a random forest model.

Acknowledgments

Jessica Hullman thanks NSF (#1907941) and Adobe.

REFERENCES

[1] N. Adrienko and G. Adrienko. Spatial generalization and aggregation of massive movement data. *IEEE Transactions on Visualization and Computer Graphics*, 2011. doi: 10.1109/TVCG.2010.44

[2] R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *IEEE Symposium on Information Visualization, 2005*, InfoVis '05. IEEE Computer Society, 2005. doi: 10.1109/INFVIS.2005.1532136

[3] T. Blascheck, L. Besançon, A. Bezerianos, B. Lee, and P. Isenberg. Glanceable visualization: Studies of data comparison performance on smartwatches. *IEEE Transactions on Visualization and Computer Graphics*, 2019. doi: 10.1109/TVCG.2018.2865142

[4] T. Blascheck. and P. Isenberg. A replication study on glanceable visualizations: Comparing different stimulus sizes on a laptop computer. In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3: IVAPP,*. INSTICC, SciTePress, 2021. doi: 10.5220/0010328501330143

[5] M. Böttinger. *Reaching Broad Audiences from a Research Institute Setting*. Springer International Publishing, 2020. doi: 10.1007/978-3-030-34444-3_17

[6] M. Brehmer, B. Lee, P. Isenberg, and E. K. Choe. A comparative evaluation of animation and small multiples for trend visualization on mobile phones. *IEEE Transactions on Visualization and Computer Graphics*, 2020. doi: 10.1109/TVCG.2019.2934397

[7] M. Brehmer and T. Munzner. A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 2013. doi: 10.1109/TVCG.2013.124

[8] G. Brewka, T. Eiter, and M. Truszczyński. Answer set programming at a glance. *Commun. ACM*, 2011. doi: 10.1145/2043174.2043195

[9] A. Brychtová and A. Çöltekin. The effect of spatial distance on the discriminability of colors in maps. *Cartography and Geographic Information Science*, 2017. doi: 10.1080/15230406.2016.1140074

[10] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05. ACM, 2005. doi: 10.1145/1102351.1102363

[11] M. Cappellari, R. M. McDermid, K. Alatalo, L. Blitz, M. Bois, F. Bournaud, M. Bureau, A. F. Crocker, R. L. Davies, T. A. Davis, P. T. de Zeeuw, P.-A. Duc, E. Emsellem, S. Khochfar, D. Krajnović, H. Kuntschner, R. Morganti, T. Naab, T. Oosterloo, M. Sarzi, N. Scott, P. Serra, A.-M. Weijmans, and L. M. Young. The atlas^{3D} project - xx. mass-size and mass-σ distributions of early-type galaxies: bulge fraction drives kinematics, mass-to-light ratio, molecular gas fraction and stellar initial mass function. *MNRAS*, 2013. doi: 10.1093/mnras/stt644

[12] W. S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 1979. doi: 10.1080/01621459.1979.10481038

[13] K. A. Cook and J. J. Thomas. Illuminating the path: The research and development agenda for visual analytics. Technical report, National Visualization and Analytic Center, 2005. https://www.hsdl.org/?abstract&did=485291.

[14] Q. Cui, M. Ward, E. Rundensteiner, and J. Yang. Measuring data abstraction quality in multiresolution visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 2006. doi: 10.1109/TVCG.2006.161

[15] Z. Cui, S. K. Badam, M. A. Yalçin, and N. Elmqvist. Datasite: Proactive visual data exploration with computation of insight-based recommendations. *Information Visualization*, 2019. doi: 10.1177/1473871618806555

[16] Ç. Demiralp, M. S. Bernstein, and J. Heer. Learning perceptual kernels for visualization design. *IEEE Transactions on Visualization and Computer Graphics*, 2014. doi: 10.1109/TVCG.2014.2346978

[17] Ç. Demiralp, P. J. Haas, S. Parthasarathy, and T. Pedapati. Foresight: Recommending visual insights. *Proceedings of the VLDB Endowment*, 2017. doi: 10.14778/3137765.3137813

[18] E. Di Giacomo, W. Didimo, G. Liotta, and F. Montecchiani. Network visualization retargeting. In *2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA)*, 2015. doi: 10.1109/IISA.2015.7388095

[19] N. Elmqvist and J. Fekete. Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines. *IEEE Transactions on Visualization and Computer Graphics*, 2010. doi: 10.1109/TVCG.2009.84

[20] M. D. Fairchild. *Color appearance models*. Wiley Blackwell, 2004.

[21] M. Gebser, R. Kaminski, B. Kaufmann, and T. Schaub. Clingo = ASP + control: Preliminary report. 2014. https://arxiv.org/abs/1405.3694.

[22] M. Gebser, B. Kaufmann, R. Kaminski, M. Ostrowski, T. Schaub, and M. Schneider. Potassco: The potsdam answer set solving collection. *AI Commun.*, 2011. doi: 10.3233/AIC-2011-0491

[23] A. Gelman and A. Guzey. Statistics as squid ink: How prominent researchers can get away with misrepresenting data. *CHANCE*, 2020. doi: 10.1080/09332480.2020.1754069

[24] J. Harper and M. Agrawala. Deconstructing and restyling d3 visualizations. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14. ACM, 2014. doi: 10.1145/2642918.2647411

[25] J. Harper and M. Agrawala. Converting basic d3 charts into reusable style templates. *IEEE Transactions on Visualization and Computer Graphics*, 2017. doi: 10.1109/TVCG.2017.2659744

[26] J. Hasell. Which countries have protected both health and the economy in the pandemic?, 2020. https://ourworldindata.org/covid-health-economy Last accessed: March 1, 2021.

[27] J. Heer and M. Bostock. Crowdsourcing graphical perception: Using mechanical turk to assess visualization design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10. ACM, 2010. doi: 10.1145/1753326.1753357

[28] J. Heer, N. Kong, and M. Agrawala. Sizing the horizon: The effects of chart size and layering on the graphical perception of time series visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09. ACM, 2009. doi: 10.1145/1518701.1518897

[29] R. Herbrich. Support vector learning for ordinal regression. In *Proceedings of the 9th International Conference on Artificial Neural Networks*, ICANN '99. Institution of Engineering and Technology, 1999. doi: 10.1049/cp:19991091

[30] J. Hoffswell, W. Li, and L. Zhicheng. Techniques for flexible responsive visualization design. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20. ACM, 2020. doi: 10.1145/3313831.3376777

[31] J. M. Hofman, D. G. Goldstein, and J. Hullman. How visualizing inferential uncertainty can mislead readers about treatment effects in scientific results. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20. ACM, 2020. doi: 10.1145/3313831.3376454

[32] K. Hu, M. A. Bakker, S. Li, T. Kraska, and C. Hidalgo. Vizml: A machine learning approach to visualization recommendation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19. ACM, 2019. doi: 10.1145/3290605.3300358

[33] J. Hullman and N. Diakopoulos. Visualization rhetoric: Framing effects in narrative visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2011. doi: 10.1109/TVCG.2011.255

[34] J. Hullman, S. Drucker, N. H. Riche, B. Lee, D. Fisher, and E. Adar. A deeper understanding of sequence in narrative visualization. *IEEE Transactions on visualization and computer graphics*, 2013. doi: 10.1109/TVCG.2013.119

[35] P. Isenberg, P. Dragicevic, W. Willett, A. Bezerianos, and J. Fekete. Hybrid-image visualization for large viewing environments. *IEEE Transactions on Visualization & Computer Graphics*, 2013. doi: 10.1109/TVCG.2013.163

[36] K. G. Jamieson and R. D. Nowak. Active ranking using pairwise comparisons. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, eds., *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2011. https://proceedings.neurips.cc/paper/2011/file/6c14da109e294d1e8155be8aa4b1ce8e-Paper.pdf.

[37] H. Jänicke and M. Chen. A salience-based quality metric for visualization. *Computer Graphics Forum*, 2010. doi: 10.1111/j.1467-8659.2009.01667.x

[38] W. Javed and N. Elmqvist. Exploring the design space of composite visualization. In *2012 IEEE Pacific Visualization Symposium*, PacificVis '12. IEEE Computer Society, 2012. doi: 10.1109/PacificVis.2012.6183556

[39] S. Johansson and J. Johansson. Interactive dimensionality reduction through user-defined combinations of quality metrics. *IEEE Transactions on Visualization and Computer Graphics*, 2009. doi: 10.1109/TVCG.2009.153

[40] G. T. Johnson and S. Hertig. A guide to the visual analysis and communication of biomolecular structural data. *Nature Reviews Molecular Cell Biology*, 2014. doi: 10.1038/nrm3874

[41] C. Kelleher and T. Wagener. Ten guidelines for effective data visualization in scientific publications. *Environmental Modelling & Software*, 2011. doi: 10.1016/j.envsoft.2010.12.006

[42] M. G. Kendall. *Rank correlation methods*. Griffin, 1948.

[43] H. Kim, D. Moritz, and J. Hullman. Design patterns and trade-offs in

authoring responsive visualization. *Computer Graphics Forum*, 2021. doi: 10.1111/cgf.14321

[44] Y. Kim and J. Heer. Assessing effects of task and data distribution on the effectiveness of visual encodings. *Computer Graphics Forum*, 2018. doi: 10.1111/cgf.13409

[45] Y. Kim, K. Wongsuphasawat, J. Hullman, and J. Heer. Graphscape: A model for automated reasoning about visualization similarity and sequencing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17. ACM, 2017. doi: 10.1145/3025453.3025866

[46] G. Kindlmann and C. Scheidegger. An algebraic process for visualization design. *IEEE Transactions on Visualization and Computer Graphics*, 2014. doi: 10.1109/TVCG.2014.2346325

[47] H. Lin, D. Moritz, and J. Heer. Dziban: Balancing agency & automation in visualization design via anchored recommendations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20. ACM, 2020. doi: 10.1145/3313831.3376880

[48] Y. Luo, X. Qin, N. Tang, and G. Li. Deepeye: Towards automatic data visualization. In *2018 IEEE 34th International Conference on Data Engineering*, ICDE '18. IEEE Computer Society, 2018. doi: 10.1109/ICDE.2018.00019

[49] Y. Ma, A. K. H. Tung, W. Wang, X. Gao, Z. Pan, and W. Chen. Scatternet: A deep subjective similarity model for visual analysis of scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1562–1576, 2020. doi: 10.1109/TVCG.2018.2875702

[50] 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 Transactions on Visualization and Computer Graphics*, 2018. doi: 10.1109/TVCG.2018.2865240

[51] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 2011. `http://jmlr.org/papers/v12/pedregosa11a.html`.

[52] Z. Qu and J. Hullman. Evaluating visualization sets: Trade-offs between local effectiveness and global consistency. In *Proceedings of the Sixth Workshop on Beyond Time and Errors on Novel Evaluation Methods for Visualization*, BELIV '16. ACM, 2016. doi: 10.1145/2993901.2993910

[53] M. Roser. Human development index (hdi), 2014. `https://ourworldindata.org/human-development-index`. Last accessed: March 1, 2021.

[54] M. Roser and E. Ortiz-Ospina. Income inequality, 2013. `https://ourworldindata.org/income-inequality` Last accessed: March 1, 2021.

[55] B. Saket, A. Srinivasan, E. D. Ragan, and A. Endert. Evaluating interactive graphical encodings for data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2018. doi: 10.1109/TVCG.2017.2680452

[56] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, 2017. doi: 10.1109/TVCG.2016.2599030

[57] A. Satyanarayan, R. Russell, J. Hoffswell, and J. Heer. Reactive vega: A streaming dataflow architecture for declarative interactive visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2016. doi: 10.1109/TVCG.2015.2467091

[58] E. Segel and J. Heer. Narrative visualization: Telling stories with data. *IEEE Transactions on Visualization and Computer Graphics*, 2010. doi: 10.1109/TVCG.2010.179

[59] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x

[60] S. Smart and D. A. Szafir. Measuring the separability of shape, size, and color in scatterplots. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19. ACM, 2019. doi: 10.1145/3290605.3300899

[61] A. Srinivasan, S. M. Drucker, A. Endert, and J. Stasko. Augmenting visualizations with interactive data facts to facilitate interpretation and communication. *IEEE Transactions on Visualization and Computer Graphics*, 2019. doi: 10.1109/TVCG.2018.2865145

[62] S. S. Stevens. On the psychophysical law. *Psychological review*, 1957. doi: 10.1037/h0046162

[63] S. S. Stevens. *Psychophysics: Introduction to its perceptual, neural and social prospects*. Routledge, 2 ed., 2017.

[64] D. A. Szafir. Modeling color difference for visualization design. *IEEE Transactions on Visualization and Computer Graphics*, 2018. doi: 10.1109/TVCG.2017.2744359

[65] B. Tang, S. Han, M. L. Yiu, R. Ding, and D. Zhang. Extracting top-k insights from multi-dimensional data. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17. ACM, 2017. doi: 10.1145/3035918.3035922

[66] UW Interactive Data Lab. Vega: View api, 2017. `https://vega.github.io/vega/docs/api/view/#view_getState`. Last accessed: June 18, 2021.

[67] S. van der Walt and N. Smith. mpl colormaps, 2015. `https://bids.github.io/colormap/`. Last accessed: March 1, 2021.

[68] R. Veras and C. Collins. Discriminability tests for visualization effectiveness and scalability. *IEEE Transactions on Visualization and Computer Graphics*, 2020. doi: 10.1109/TVCG.2019.2934432

[69] C. Villani. *The Wasserstein distances*. Springer Berlin Heidelberg, 2009. doi: 10.1007/978-3-540-71050-9_6

[70] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 2020. doi: 10.1038/s41592-019-0686-2

[71] Y. Wang, F. Han, L. Zhu, O. Deussen, and B. Chen. Line graph or scatter plot? automatic selection of methods for visualizing trends in time series. *IEEE Transactions on Visualization and Computer Graphics*, 2018. doi: 10.1109/TVCG.2017.2653106

[72] M. Wattenberg and D. Fisher. A model of multi-scale perceptual organization in information graphics. In *IEEE Symposium on Information Visualization 2003*, InfoVis '03. IEEE Computer Society, 2003. doi: 10.1109/INFVIS.2003.1249005

[73] M. Wattenberg and D. Fisher. Analyzing perceptual organization in information graphics. *Information Visualization*, 2004. doi: 10.1057/palgrave.ivs.9500070

[74] A. Wu, W. Tong, T. Dwyer, B. Lee, P. Isenberg, and H. Qu. Mobilevisfixer: Tailoring web visualizations for mobile phones leveraging an explainable reinforcement learning framework. *IEEE Transactions on Visualization and Computer Graphics*, 2020. doi: 10.1109/TVCG.2020.3030423

[75] A. Wu, L. Xie, B. Lee, Y. Wang, W. Cui, and H. Qu. Learning to automate chart layout configurations using crowdsourced paired comparison. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21. ACM, 2021. doi: 10.1145/3411764.3445179

[76] Y. Wu, X. Liu, S. Liu, and K. Ma. Visizer: A visualization resizing framework. *IEEE Transactions on Visualization and Computer Graphics*, 2013. doi: 10.1109/TVCG.2012.114

[77] E. Zgraggen, Z. Zhao, R. Zeleznik, and T. Kraska. Investigating the effect of the multiple comparisons problem in visual analysis. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18. ACM, 2018. doi: 10.1145/3173574.3174053