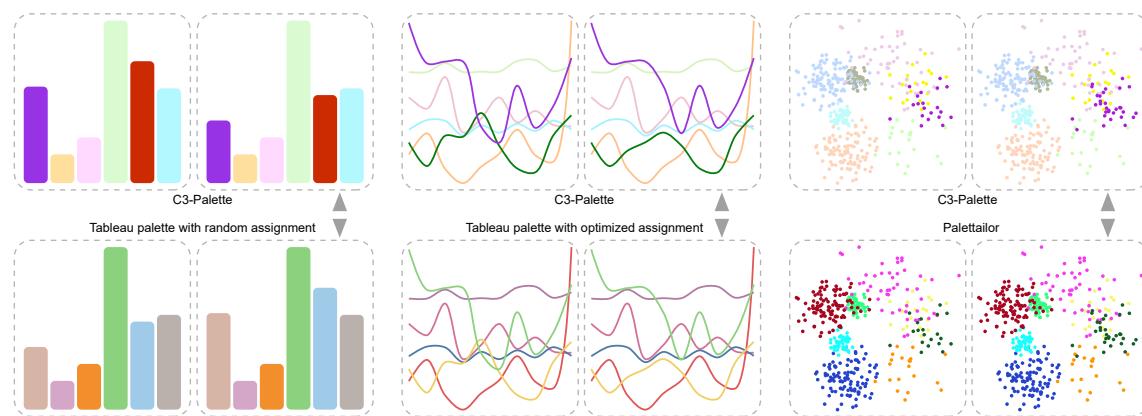


1 **C³-palette: Co-saliency based Colorization for Comparing Categorical
2 Visualizations**
3

4 ANONYMOUS AUTHOR(S)
5



22 Fig. 1. Results for different types of categorical data visualizations: (left) C³-palette versus Tableau palette with random assignment;
23 (center) C³-palette versus Tableau palette with optimal assignment; (right) C³-palette versus Palettailor [29]. Our co-saliency methods
24 (top) can highlight the changed classes while maintaining discrimination of classes.

25 Visual comparison within juxtaposed views is an essential part of interactive data analysis. In this paper, we propose a co-saliency
26 model to characterize the most co-salient features among juxtaposed labeled data visualizations while maintaining class discrimination
27 in the individual visualizations. Based on this model, we present a comparison-driven color design framework, enabling the automatic
28 generation of colors that maximizes co-saliency among juxtaposed visualizations for better identifying items with the largest magnitude
29 change between two data sets. We conducted two online controlled experiments to compare our colorizations of bar charts and
30 scatterplots with results produced by existing single view-based color design methods. We further present an interactive system and
31 conduct a case study to demonstrate the usefulness of our method for comparing juxtaposed line charts. The results show that our
32 approach is able to generate high quality color palettes in support of visual comparisons of juxtaposed categorical visualizations.
33
34

35 CCS Concepts: • Human-centered computing → Information visualization.
36

37 Additional Key Words and Phrases: Color Palette, Visual Comparison, Multi-Class, Juxtaposition
38

39 **ACM Reference Format:**

40 ANONYMOUS AUTHOR(S). 2018. C³-palette: Co-saliency based Colorization for Comparing Categorical Visualizations. In *Woodstock*
41 '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY. ACM, New York, NY, USA, 22 pages. <https://doi.org/10.1145/1122445.1122456>

45 Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not
46 made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components
47 of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to
48 redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

49 © 2018 Association for Computing Machinery.

50 Manuscript submitted to ACM

53 1 INTRODUCTION

54 Comparison is an indispensable task in data analysis and visualization. It often involves searching for categories (classes)
 55 with large or small changes among multiple categorical datasets. Comparison are usually achieved through juxtaposition
 56 of multiple categorical visualizations [13, 30] such as bar charts, line charts or multi-class scatterplots, where each
 57 category is commonly encoded by a unique color. While color codings are known to play an important role in helping
 58 viewers see differences between juxtaposed views [1, 13, 40], there is no color design scheme that optimizes visual
 59 comparisons, especially for the task of identifying the largest differences between charts [34].
 60

61 A typical scenario would be a market analyst who uses comparisons to investigate the performance of a company
 62 across different countries over the last two years. S/he first would create a scatterplot for each year by showing the
 63 annual revenue and profit of each product by colorizing each point of the plot with a country label. After finding the
 64 two countries with the largest changes, s/he then examines the annual and monthly profits of various products in these
 65 countries with bar and line charts. Using the product name for color encoding, s/he is able to efficiently search the
 66 product with the largest differences from side-by-side shown bar and line charts.
 67

68 The most common way to colorize juxtaposed views is to manually find a color mapping for a selected view while
 69 judging how well it fits to the other views. Such a trial and error procedure might converge to a desirable color
 70 mapping; however, the needed effort significantly increases with the numbers of classes and views. Although existing
 71 automated color selection approaches [6, 29, 41] allow to alleviate the effort for single view colorizations, the obtained
 72 color mapping might not be able to clearly reveal similarities or differences among multiple views. For example, the
 73 assignment of the Tableau palette for maximizing class separability (cf. [41]) in Fig. 1(top middle) creates a visualization
 74 with better class discrimination, but the changed time series (the green and red ones in middle bottom) are hard to
 75 identify. Although such classes of interest can be highlighted by fading out background classes using alpha blending, it
 76 inevitably introduces visual ambiguity for overlapping classes [3] and potentially leads to a poor class separation.
 77 As far as we know, few existing visualization-oriented color selection tools (e.g., ColorBrewer [16] or Palettailor [29])
 78 allow for colorizing multi-view visualizations, let alone supporting comparisons in juxtaposed views.
 79

80 To fill this gap, we propose a comparison-driven color palette generation framework, which automatically generates
 81 appropriate color mappings for efficiently searching the largest delta from one categorical visualization to another.
 82 To achieve this goal, we propose a co-saliency model to characterize the most salient features among juxtaposed
 83 categorical visualizations that are likely to attract visual attention. We borrow the idea from the concept of image
 84 co-saliency [20], which was originally designed for summarizing salient differences between two similar natural images.
 85 In line with this, we devise our co-saliency model for easily identifying important features (e.g., changed classes) from
 86 juxtaposed categorical visualizations while maximizing the visual discrimination of classes in individual visualizations.
 87 It is achieved by fusing class importance between visualizations and class contrast within visualizations. The class
 88 contrast is based on perceptual separability with neighboring classes and with the background [41], while the class
 89 change is measured by combining point position change and point number change of each class, where the position
 90 change is quantified by using a perceptual distance metric, Earth Mover's Distance (EMD) [36]. That is, the classes with
 91 large importance and small class separabilities (strong overlap with another classes) are more co-salient, while the ones
 92 with small importance or large separabilities (more compact) being less co-salient.
 93

94 By integrating our co-saliency model into existing categorical data colorization tools [29], we can automatically
 95 generate color mappings that maximize co-saliency among juxtaposed visualizations. The resulted color mapping scheme
 96 makes the classes with large importance pop out from the context and attract viewers' attention, while maximizing the
 97

105 perceptual separability between classes in individual visualizations. By doing so, the major issue [39] of the juxtaposition
 106 is that human have limited visual memory is greatly alleviated and the visual search can be done with less cognitive
 107 cost [17]. The top of Fig. 1 shows the results generated by our colorization method, where the changed classes are pop
 108 out and easier to be spot than the ones in the bottom of Fig. 1. We can see that our results are similar to the ones of
 109 alpha blending, but still maintains the separability between classes due to different hues.
 110

111 We evaluated our approach through carefully designed bar charts and scatterplots by comparing our colorized
 112 results with the ones produced by the state-of-the-art palettes (e.g.,Tableau [38] and Palettailor [29]). For bar charts,
 113 we replicated the experimental setting of Ondov et al. [34] but only performed the task of identifying a maximum
 114 delta from horizontal juxtaposed bar charts. Next, we carried out the studies for scatterplots with the multi-class
 115 scatterplots generated by Lu et al. [29], whose counterparts are generated by changing properties. (point number, point
 116 position) of several randomly selected classes. We first conducted a pilot study to verify the validity of our experiment
 117 setting and then ran two online studies to investigate how well our generated palettes help users to identify changed
 118 classes between two scatterplots and the discrimination task of counting class numbers in single scatterplots. Last, we
 119 conducted a case study to show how our system helps for juxtaposed comparison of multiple line charts. The results
 120 show that our approach is able to produce color mapping optimized for supporting comparison and aligned with the
 121 state-of-the-art palettes in maximizing perceptual class separability.
 122

123 We furthermore develop a web-based color design tool, C³-palette ¹ which is named by Co-saliency based Colorization
 124 for Comparing categorical visualizations, using coordinated views for users to explore the relationship among multiple
 125 data with different color mapping schemes. Inspired by volume visualization [?], we provide a transfer function view
 126 for users to highlight different classes of interest. The main contributions of this paper are as follows:
 127

- 128 • We propose a multi-class data visualization co-saliency model for measuring the importance of each data item
 129 shown in juxtaposed visualizations and use this metric to automatically generate color mapping schemes for
 130 effective comparisons;
- 131 • We evaluate the effectiveness of the resulting color mapping schemes in supporting both visual comparison and
 132 visual discriminability with three online user studies and a case study (Section 4); and
- 133 • We provide an interactive tool that show how our approach can be used for helping visual comparison of multiple
 134 labeled data.

140 2 RELATED WORK

141 We begin by reviewing previous work related to visual comparison, color design for visualization, and visual saliency/co-
 142 saliency.
 143

144 2.1 Visual Comparison

145 Visual comparison is an essential part of interactive data analysis, which is regarded as a high-level “compound task.”
 146 Gleicher et al. [14] provide a systematic review of techniques developed for better supporting comparison and summarize
 147 three basic layout designs for comparative visualization, including *juxtaposition*, *superposition* and *explicit encoding*.
 148 Among them, juxtaposition places different datasets in different views without any change to the original visualization
 149 design and thus it is commonly used in many applications [1, 28, 33]. However, it often causes cognitive burden because
 150

151 ¹<https://c3-palette.github.io/>

157 users need to maintain a mental image of one view for comparing with another view [30]. Recently, Ondov et al. [34]
 158 and Jardine et al. [22] evaluated the perceptual effectiveness of different layouts for bar charts comparison with a few
 159 low-level tasks, which show that juxtaposition is less effective in some tasks like finding “biggest delta between items.”
 160 Accordingly, Gleicher et al. [14] and L’Yi et al. [30] both suggested to carefully design visual encoding for improving its
 161 effectiveness. Our method facilitates visual comparison of categorical data by improving the visual search with the
 162 pop-out effect [10] induced by our proposed color mapping scheme.
 163

165 2.2 Color Design

166 For a complete review of color design for visualization, we refer readers to survey papers [40, 45]. We limit our discussion
 167 to the techniques related to color design for categorical data visualization including color mapping optimization, color
 168 palette generation, and color design for multi-view visualization.
 169

170 **Color Mapping Optimization.** Mapping each class to a proper color selected from the given palette is particularly
 171 helpful for categorical data visualization. A few different factors have been used for guiding the search of such mappings.
 172 For example, Lin et al. [27] proposed to optimize the compatibility between the class semantics and the assigned colors.
 173 Setlur and Stone [37] produced better results by using co-occurrence measures of color name frequencies. For the
 174 classes without clear semantics, Hurter et al. [18] suggested to maximize perceptual color differences among close lines
 175 of a metro map. Kim et al. [23] incorporated color aesthetics and color contrast into the optimization of color assignment
 176 for image segments. Recently, Wang et al. [41] proposed to maximize class discriminability based on color-based class
 177 separability, which takes into account spatial relationships between classes and the contrast with background color.
 178 Once an assignment is specified, the color for each class can be further optimized to better serve different purposes,
 179 such as reducing power consumption of displays [7], improving the accessibility of visualizations for visual impaired
 180 users [31], and better class discrimination [26]. Almost all these methods aim to generate effective visualizations for
 181 single data, whereas our goal is to efficiently visualize salient class differences across multiple similar datasets with the
 182 same label information. One example is the instances of the same datasets evolving over time.
 183

184 **Color Palette Generation.** To have an appropriate categorical color palette, the commonly used approach is to
 185 select from a library of carefully designed palettes provided by online tools (e.g. ColorBrewer [16]). Colorgorical [15]
 186 further allows users to customize color palettes by generating palettes based on user-specified discriminability and
 187 preference importance. Chen et al. [6] suggested to directly search proper colors in CIELAB space for maximizing class
 188 discrimination in multi-class scatterplots. Yet, it cannot find enough colors with large color differences, because of
 189 leaving out L* channel in the optimization. Recently, Palettailor [29] takes a further step that can automatically generate
 190 categorical palettes for different types of charts, such as scatterplots, line and bar charts. All the aforementioned methods
 191 deal with single data, while our work focuses on visual comparison of multiple similar labeled datasets with some
 192 changed instances.
 193

194 **Multi-view Color Design.** Multi-view visualizations are commonly used in multivariate analysis. Although a few
 195 design guidelines [42] have been proposed for constructing multi-view visualizations, few of them are related to color
 196 design. Qu et al. [35] recommended a set of color consistency constraints across views. Among them, the high level
 197 constraint that the same data field should be encoded in the same way is related to our studied comparative visualization.
 198 Namely, all juxtaposed views should have the same color mapping scheme and a good scheme can help for seeing the
 199 differences between views. However, few work has been done for finding such schemes. The only exception is comparing
 200 multiple continuous scalar fields [40] with an improved global color map by merging overlapping value ranges in
 201

209 different datasets. Our work is the first to generate appropriate color mapping for comparing multiple categorical
 210 visualizations.
 211

212 2.3 Visual Saliency & Co-saliency

213 Here we briefly review the visual saliency model developed for visualizations and image co-saliency models.
 214

215 **Saliency for Visualization.** The human visual system enables viewers to concentrate on salient regions of an image
 216 while ignoring the others. It is guided by two major factors [8]: pre-attentive, bottom-up attention based on visual
 217 features (e.g., color, intensity and edges) and task-driven, top-down attention based on prior knowledge. A numerous of
 218 saliency models [4] have been developed to mimic bottom-up attention mechanism in computer vision community.
 219 Most of them model image saliency as the contrast of image regions to their surroundings with low level features.
 220 Among them, the most influential one is the Itti model [19], which computes image saliency with central surrounded
 221 differences. Kim et al. [24] tailored this model to increase the visual saliency of selected regions of a volume dataset.
 222 Jänicke and Chen [21] employed this model [19] to define a quality metric for evaluating visualizations. Recently,
 223 Matzen et al. [32] evaluated a variety of saliency models on a large visualization dataset and explored why these
 224 models work poorly for visualization images. One major reason is that visualizations are often created for specific goals,
 225 whereas existing models are based on the bottom-up attention. To overcome these weaknesses, they proposed a data
 226 visualization saliency (DVS) model by incorporating meaningful high-level text features into Itti's model. However, this
 227 model is not designed on the class-level and cannot be directly used for categorical visualizations.
 228

229 **Image Co-Saliency.** Unlike single image based saliency model, the co-saliency model estimates the saliency (importance)
 230 of each pixel within the context of multiple related images. Jacobs et al. [20] developed the first co-saliency model
 231 for highlighting the most salient differences between two compared images. Later, this concept has been extended for
 232 discovering common and salient objects/foregrounds from image collections [44]. Inspired by the original model [20],
 233 our work attempts to design an appropriate color mapping for visualizing the most co-salient features among juxtaposed
 234 labeled data visualizations. Following their findings that the co-salient features can be effectively characterized by
 235 fusing image changes and single image contrast together, our co-saliency model relies on two factors: the class contrast
 236 in individual views and global features from in-between views (e.g., class structure changes).
 237

238 3 CO-SALIENCY BASED COLOR DESIGN

239 Given multiple categorical visualizations with the same class labels (or a subset thereof), each visualization has K
 240 classes and n_j data items $\{\mathbf{x}_1^j, \dots, \mathbf{x}_{n_j}^j\}$, where each \mathbf{x}_t^j has a label $l(\mathbf{x}_t^j)$ and the i -th class (with n_i^j data points) consists
 241 of $\{\mathbf{x}_{i,1}^j, \dots, \mathbf{x}_{i,n_i^j}^j\}$, $i \in \{1, \dots, m\}$. For standard bar charts and line charts, K is equal to n_j but often is smaller than n_j
 242 in scatterplots. All visualizations use the same background color c_b and the same color mapping scheme $\tau : L \mapsto c$. Our
 243 goal is to find the best mapping τ that supports effective comparison of multiple categorical scatterplots.
 244

245 In line with the design requirements of natural image comparison and categorial data visualization [13, 20, 29], our
 246 problem is formulated based on the following three design requirements:
 247

- 248 (i) **DR1:** highlighting the most concerned classes between visualizations as much as possible for an efficient
 249 comparison;
- 250 (ii) **DR2:** maximizing the visual discrimination between classes in individual visualizations for an efficient exploration
 251 of multi-class data; and
- 252 (iii) **DR3:** providing flexible interactions for the exploration of relationships among the compared datasets.
 253

261 Although visual comparison is an essential part of interactive data analysis, most of the existing colorization techniques [15, 29] attempt to meet DR2. The key challenge in meeting DR1 is that we need a proper model to characterize the
 262 most salient features in multiple visualizations. To address this issue, we propose a categorical visualization co-saliency
 263 model that calculates the saliency of each data item in the context of other similar visualizations. Integrating this
 264 model into the objective of the state-of-the-art color mapping generation framework [29], we can generate proper color
 265 mappings to highlight salient differences between juxtaposed categorical visualizations while fostering better visual
 266 discrimination of classes.
 267

270

271

272

273

274 3.1 Co-saliency for Multi-class Scatterplots

275

276 Following the definition of image co-saliency [20], we model the class co-saliency with two factors: class importance
 277 between visualizations and class contrast within visualizations. The class importance describes how much each class
 278 should stand out from the visualization. describes how much each class stands out from neighboring classes and the
 279 background, which is similar to perceptual class separability [2, 41]. Hence, we define two types of class contrasts: a
 280 local contrast with neighboring classes and a background contrast with the background.
 281

282

283 Since point-based representation is general in 2D visualizations, we use scatterplots to illustrate our method.
 284 Analogous to bottom-up image co-saliency models [11, 20], the co-saliency of the i th class is defined as the product
 285 between class importance and class contrast score to emphasize the target class, and the co-saliency for M classes:
 286

287

288

$$E_{CoS} = \sum_i \left(\sum_j \frac{1}{n_i^j} (\lambda \alpha_i^j \exp(\theta_i) + (1 - \lambda) \beta_i^j f(\theta_i)) \right) \quad (1)$$

289

290 where θ_i is the importance of the i th class, n_i^j is the number of point of the i th class in the j th scatterplot, α_i^j is the local
 291 contrast with neighboring classes of the i th class in the j th scatterplot, β_i^j is the contrast to the background, and λ is a
 292 weight between them. The weight $1/n_i^j$ is used to alleviate class imbalance so that the classes with small numbers of
 293 points and large changes can be highlighted.
 294

295

296 To better support DR1, we apply an exponential function to enlarge the weight of class structure changes, while
 297 using a piecewise function to weight the background contrast:
 298

299

300

$$f(\theta_i) = \begin{cases} \exp(\theta_i) & \text{if } \theta_i > \kappa \\ -\exp(\theta_i) & \text{else} \end{cases} \quad (2)$$

301

302 κ is a user-specified threshold with the default zero. The reason for the two different weighting schemes is that classes
 303 with less or no changes might be treated as the background by viewers [44]. To suppress the saliency of such classes,
 304 we introduce a negative importance for them.
 305

306

307 **Local Contrast.** Given the j th scatterplot, we define the local class contrast based on the α -shape based point distinctness [29]. For each data point \mathbf{x}_t^j , we define its point distinctness value as:
 308

309

310

311

312

$$\gamma(\mathbf{x}_t^j) = \frac{1}{|\Omega_t^j|} \sum_{\mathbf{x}_p^j \in \Omega_t^j} \frac{\Delta\epsilon(\tau(l(\mathbf{x}_t^j)), \tau(l(\mathbf{x}_p^j)))}{d(\mathbf{x}_t^j, \mathbf{x}_p^j)},$$

313 where Ω_t^j is set of k nearest neighbors of \mathbf{x}_t^j , $\tau(l(\mathbf{x}_p^j))$ is the color of \mathbf{x}_p^j , d is the Euclidean distance and $\Delta\epsilon$ is CIELAB
 314 color distance. For the i th class, its local contrast is the sum of all points with the same class label in the scatterplot:
 315

$$\phi_i^j = \frac{1}{n_j} \sum_p^{n_j} \gamma(\mathbf{x}_p^j) \delta(l(\mathbf{x}_p^j), i) \quad (3)$$

316 where $\delta(l(\mathbf{x}_p^j), i)$ is one if the class label $l(\mathbf{x}_p^j)$ is i and else zero.
 317

318 If one class overlaps with different classes, the local contrast value is high and the value is small for a well separated
 319 class. Hence, the black class in two scatterplots shown in Fig. 2(a) both has a low contrast value (see Fig. 2(b)) and the
 320 cyan class has a large value.
 321

322 **Background Contrast.** The contrast with the background is based on the point non-separability [41] $\rho(\mathbf{x}_t^j)$, defined as
 323 the difference between two separation degrees:
 324

$$\rho(\mathbf{x}_t^j) = b(\mathbf{x}_t^j) - a(\mathbf{x}_t^j). \quad (4)$$

325 where $b(\mathbf{x}_t^j)$ and $a(\mathbf{x}_t^j)$ are between-class separation degree and within-class separation degree of \mathbf{x}_t^j . They are defined
 326 as the sum of the weighted color differences from its neighborhood with the same and different class labels:
 327

$$a(\mathbf{x}_t^j) = \frac{1}{|\Omega_t^j|} \sum_{\mathbf{x}_p^j \in \Omega_t^j} \frac{\delta(l(\mathbf{x}_t^j), l(\mathbf{x}_p^j)) \Delta\epsilon(\tau(l(\mathbf{x}_t^j)), \mathbf{c}_b)}{d(\mathbf{x}_t^j, \mathbf{x}_p^j)}, \quad b(\mathbf{x}_t^j) = \frac{1}{|\Omega_t^j|} \sum_{\mathbf{x}_p^j \in \Omega_t^j} \frac{1 - \delta(l(\mathbf{x}_t^j), l(\mathbf{x}_p^j)) \Delta\epsilon(\tau(l(\mathbf{x}_t^j)), \mathbf{c}_b)}{d(\mathbf{x}_t^j, \mathbf{x}_p^j)}$$

328 When most neighbor points of \mathbf{x}_t^j have the same label as \mathbf{x}_t^j , $\rho(\mathbf{x}_t^j)$ is negative and vice versa. However, a negative
 329 $\rho(\mathbf{x}_t^j)$ makes the optimization in Eq. 1 meaningless, resulting that its belonged classes might be highlighted no matter
 330 how large the change of this class is. To address the above issue, we use an exponential function to convert $\rho(\mathbf{x}_t^j)$ to be
 331 positive while maintaining the monotonicity. Accordingly, we define the contrast to the background of the i th class as:
 332

$$\beta_i^j = \frac{1}{n_i^j} \sum_t^{n_i^j} \exp(\rho(\mathbf{x}_t^j)) \delta(l(\mathbf{x}_t^j), i). \quad (5)$$

333 As illustrated in Fig. 2(c), the well-separated classes with large color differences from the background have large
 334 background contrast like the blue and black classes, whereas the pink and cyan classes have relative large background
 335 contrast values with medium class separation.
 336

337 **Class Importance.** Class importance reflects whether a class should be highlighted or not. It can be specified by user
 338 or by some measures. In our paper, we use class change degree to represent the importance of each class as default.
 339 To quantify how users perceive class structure changes, we measure the difference between class distributions in two
 340 scatterplots with the Earth Mover's Distance (EMD) [36], a perceptual metric. Suppose the i th class with two sets of
 341 points $\mathbf{X}_i^1 = \{\mathbf{x}_{i,1}^1, \dots, \mathbf{x}_{i,n_i^1}^1\}$ and $\mathbf{X}_i^2 = \{\mathbf{x}_{i,1}^2, \dots, \mathbf{x}_{i,n_i^2}^2\}$. Taking the Euclidian distance between two points as the cost,
 342 we need to minimize the total matching cost
 343

$$H(\mathbf{X}_i^1, \mathbf{X}_i^2) = \min_{\chi} \sum_t d(\mathbf{x}_{i,t}^1, \mathbf{x}_{i,\chi(t)}^2),$$

344 which constrains an one-to-one mapping χ between points. This is the classic bipartite matching problem, which can
 345 be solved by the Hungarian method [25]. When the number of points of two sets is not equal, we further take the
 346 difference between the number of points into account. In doing so, the class change degree contains both point position
 347

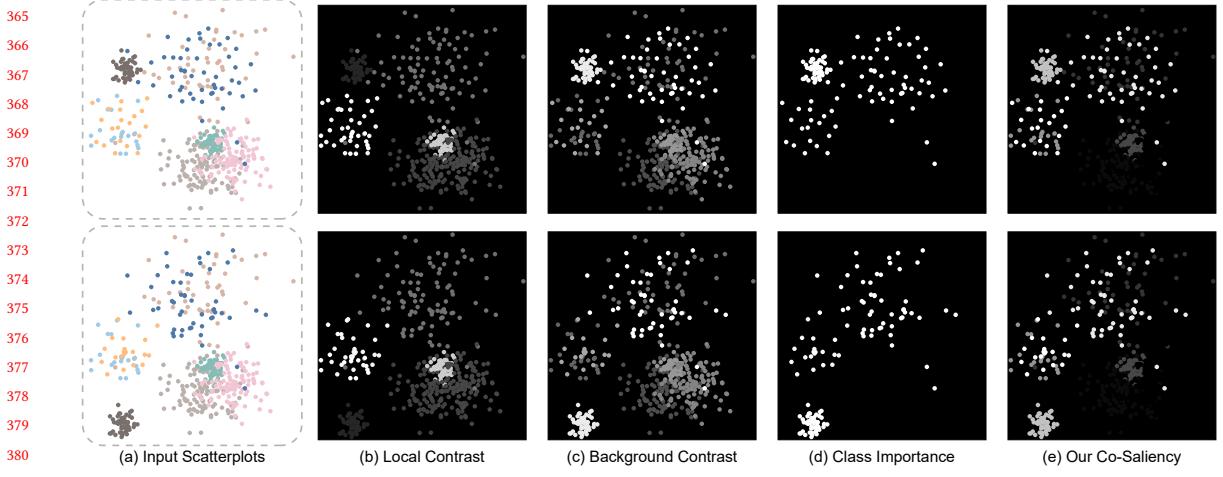


Fig. 2. Illustration of the main components for computing our co-saliency maps. For the two input scatterplots (a), our class-based co-saliency (e) is generated by fusing the local contrast (b), the background contrast (c), and the class change degree (d). The points with large brightness have large values and vice versa.

change and point number change between two sets, which is defined as:

$$\theta_i = \frac{H(\mathbf{X}_i^1, \mathbf{X}_i^2)}{\min\{n_i^1, n_i^2\}} + \nu \frac{\|n_i^1 - n_i^2\|}{\max\{n_i^1, n_i^2\}} \quad (6)$$

where both terms range within [0,1] and ν is 1.0 as the default.

Fig. 2 shows an example of two 8-class scatterplots with three classes (orange, blue and black) with changes. We can see that combining the class change degree and two contrast measures can highlight salient differences and maintain visual discrimination of classes (see Fig. 2(e)).

3.2 Co-Saliency based Palette Generation

On the basis of the co-saliency model, we meet DR1 and DR2 by co-saliency based color assignment palette generation. Taking this model in Eq. 1 as the objective of the state-of-the-art color assignment method [41], the optimal color mapping can be obtained from a given good palette. However, there are two major limitations: i) requiring users to try many palettes for selecting a good one; and ii) the design of most existing palettes is not oriented towards visual comparison so that even the best color assignment cannot provide prominent cues for this task. Fig. 3 shows an example with the Tableau-10 palette and ColorBrewer 8-class Set1 [16], respectively. We can see that both results highlight several classes with less changes (e.g., the bottom left purple class), and make the red class with the largest change hard to identify at once even it is very distinctive. Thus, we prompt users to use our co-saliency based palette generation method.

The recently proposed data-aware palette generation method [29] automatically generates discriminable and preferable palettes by maximizing the combination of three palette quality measures: point distinctness, name difference, and color discrimination. By replacing the first measure with our co-saliency model, the palette generation is formulated as an optimization problem:

$$\arg \max_{\tau} E(\tau) = \omega_0 E_{CoS} + \omega_1 E_{ND} + \omega_2 E_{CD}. \quad (7)$$

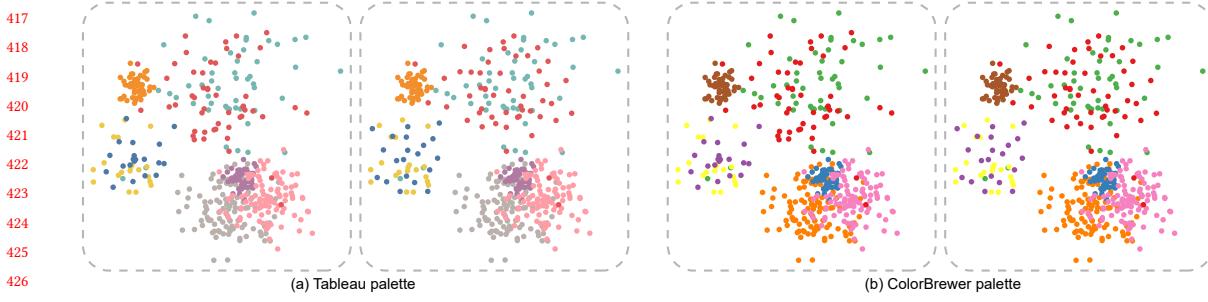


Fig. 3. The results generated by co-saliency based color assignment with the Tableau-10 palette (a) and the ColorBrewer palette (b). Most of existing palettes mainly consist of bright colors, resulting that the classes with less changes cannot be deemphasized.

which consists of a co-saliency term E_{CoS} (see Eq. 1), a name difference term E_{ND} and a color discrimination term E_{CD} , balanced by ω_0 , ω_1 and ω_2 . For more detail about E_{ND} and E_{CD} , we refer readers to [29]. By using the same optimization method as Lu et al. [29], we can generate desired colors in real time. For example, Fig. 4(b) shows an example which use same dataset within Fig. 3, improves the distinctness of the two changed classes while maintains the class separability.

3.3 Parameter Effect

Besides different weights for different terms in palette generation [29], our co-saliency model involves three parameters: the weight λ between two contrasts, the threshold for the class importance κ , and v that is related to the definition of the class change degree which is used as our default class importance. Since v is fixed in our experiments and the class importance can be specified by user, we mainly discuss the effects of λ and κ .

Balancing Weight λ . Although this parameter modulates the influence between the class contrast with neighbors and background, it offers a compromise between DR1 and DR2. As shown in Fig. 4(a), considering only the contrast to the background would have a good ‘pop out’ effect but other classes are hard to discriminate. While considering only the contrast with nearest neighbors, such as Fig. 4(d), all the classes are easy to distinguish but the changed classes are hard to find out. This is reasonable, because pre-attentive vision lets a bright saturated color region within regions of de-saturated colors “pop-out” to the viewer [17]. In our experiments, we found that setting $\lambda = 0.4$ as the default allows to simultaneously emphasize changes and preserve the discriminability between classes, see an example in Fig. 4(b).

Importance Threshold κ . The threshold κ selects the classes with large importance to be highlighted. With a default value of zero, all classes with importance value larger than zero are ensured to be highlighted. Likewise, a large κ will de-emphasize classes with a small importance. We further allow users to specify κ by interaction through the control panel.

3.4 Extension to Bar and Line Charts

Like Palettailor [29], our color mapping generation method also works for other categorical visualization types such as bar or line charts. This is achieved by treating each bar or line segment in both charts as a point and then taking the same way to compute the class contrast. Taking line charts as an example, the line segments are ordered along the time axis and we can easily build a one-to-one mapping for line segments to compute θ_i . Doing so, lines with large changes

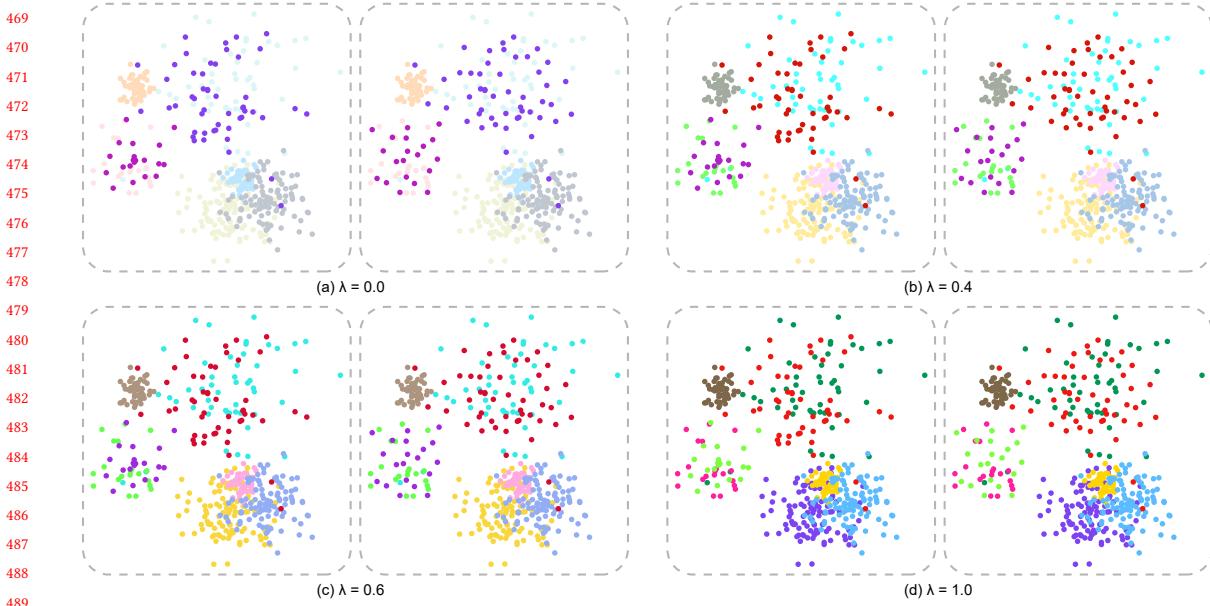


Fig. 4. Effect of λ : (a) result generated by only considering contrast to the background; (b) result generated by setting λ to 0.4; (c) result generated by setting λ to 0.6; (d) result generated by only considering contrast with nearest classes.

will be highlighted while maintaining the discriminability between multiple lines in each chart. We do the same for bar charts. An example is shown in Fig. 1, more results can be found in our supplementary material.

4 EVALUATION

We evaluated the effectiveness of our method on supporting juxtaposed visual comparisons for both scatterplots and bar charts. We conducted three online controlled experiments through Amazon Mechanical Turk (AMT) with 217 participants in total, to evaluate how well our method can support people in *observing changes* and *visual separability* for multiple categorical datasets:

- (i) *Spotting the difference task*. To evaluate how well our method can support people in *observing changes* for juxtaposed categorical scatterplots;
- (ii) *Counting class number task*. To evaluate whether our method can support the *visual separability* of classes in each individual scatterplot, which is considered fundamental to juxtaposed comparison.
- (iii) *Finding max delta task*. To evaluate how well our method can support people in *observing the largest change* for juxtaposed categorical bar charts;

Independent Variables. In each of our studies, we investigated three independent variables: colorization method, change magnitude and change type.

Colorization method: We used six different ways to colorize scatterplots, including four benchmark methods (*Random Assignment*, *Optimized Assignment*, *Alpha Blending* and *Palettaior*) and two experimental methods based on our approach (*C3-Palette Assignment*, *C3-Palette Generation*):

- C1: *Random Assignment* is randomly selecting and assigning colors from Tableau-20 palette to the classes.

- 521 • C2: *Optimized Assignment* uses the optimized assignment approach [41] for one of the two scatterplots with an
522 input of Tableau-20 color palette.
- 523 • C3: *Alpha Blending* is achieved by setting the alpha of each unchanged class to 0.5 while the changed classes
524 remain to 1.0 based on *Optimized Assignment* result. We choose 0.5 since the results also used in the discrimination
525 task.
- 526 • C4: *Palettaior* uses the method proposed by Lu et.al [29] for single scatterplot palette generation. The palette is
527 generated for one of the two scatterplots with the default settings.
- 528 • C5: *C3-Palette Assignment* uses the color assignment optimization solution(Eq. 1) based on Tableau-20.
- 529 • C6: *C3-Palette Generation* uses the unified color generation and assignment optimization method, and produced
530 the results with the default settings($\omega_0 = 1.0$, $\omega_1 = 1.0$ and $\omega_2 = 1.0$).

531 Our approach are all using the default parameters $\lambda = 0.4$ and $\kappa = 0$.

532 *Change magnitude* and *Change type*: While the colorization method is the primary independent variable to be
533 investigated, we are also interested in how the effect of different methods would vary based on the level of change
534 between the two scatterplots and the different change type of classes. Thus we first define two types of changes that a
535 class would have across multiple scatterplots: *point number* and *point position*. Then for each change type, we define
536 three levels of change magnitude calculated using Eq. 6: *small*, *medium*, and *large*. (See the next paragraph for the
537 detailed calculation.)

538 **Scatterplot Dataset Generation.** The paired scatterplot datasets used in our studies were generated as follows. First,
539 we designed a set of multi-class scatterplots, each containing 8 classes. Each class was generated using Gaussian random
540 sampling and placed randomly in a 600×600 area. Similar to [29], these classes belong to one of the four settings of
541 varying size and density: small & dense ($n = 50, \sigma = 20$), small & sparse ($n = 20, \sigma = 50$), large & dense ($n = 100, \sigma = 50$),
542 and large & sparse ($(n = 50, \sigma = 100)$).

543 Then, for each scatterplot generated above, we produced its paired scatterplot by randomly choosing one or more
544 classes and changing the positions or number of their data points. To systematically compute the changes, we defined
545 two variables: *change ratio* and *number of changed classes*. *Change ratio* defines how large the change of a type is,
546 ranging from 0 to 1; and number of changed classes defines the number of classes that are changed, ranging from 1 to 3
547 (to add different levels of difficulty). We summarize our basic idea of data generation for each change type as below.
548

- 549 • *Point number*: For each class in the original scatterplot, we calculated the new point number by multiplying the
550 original number by $(1 \pm \text{change ratio})$. An addition means to increase the point number, which was implemented
551 by generating the new points with the same distribution as the original class. Subtraction was achieved by
552 randomly deleting data points from the original class.
- 553 • *Point position*: Point position contains many types, such as class center position change and shape change. In our
554 experiment, we use the two different position changes mentioned above. For center position change, the center
555 of a class can be moved in a certain *direction* with a specific *distance*. We moved the center towards a random
556 direction by a distance calculated by multiplying a maximal change distance (400 by default) by the *change ratio*.
557 For shape change, we define the shape of a class as the bounding box of its data points. We simulated a shape
558 change of a class by modifying the density parameter of its Gaussian distribution to the opposite direction. For
559 example, a small & dense class ($n = 50, \sigma = 20$) would be changed into a small & sparse ($n = 50, \sigma = 50$) class. In
560 order to produce a new shape for a class, we first calculate the one-to-one mapping between the newly-generated
561 class and the original class using [25] and then linearly interpolated the new point between each two points
562

573 based on the *change ratio* parameter. We randomly choose one change type when disturbing the class to be
 574 changed.
 575

576 For each change type, we produced 300 candidate scatterplot pairs and then calculated the *change magnitude* for each
 577 pair, and split all pairs into three levels: *small*, *medium*, and *large*. Next, we randomly selected 2 pairs from each change
 578 magnitude level for each change type and each number of changed classes. Thus in total we used 36 paired scatterplot
 579 in each of the two studies. The detailed dataset is showed in Table. 1
 580

581 Table 1. Grouping of Datasets: 36 datasets \times 6 conditions. C: condition; G: participant group; Position Small 1: point position change
 582 with small change magnitude for 1 changed class.
 583

	C1	C2	C3	C4	C5	C6
Dataset 1: Position Small 1	G1	G2	G3	G4	G5	G6
Dataset 2: Position Small 1	G6	G1	G2	G3	G4	G5
Dataset 3: Position Small 2	G5	G6	G1	G2	G3	G4
Dataset 4: Position Small 2	G4	G5	G6	G1	G2	G3
Dataset 5: Position Small 3	G3	G4	G5	G6	G1	G2
Dataset 6: Position Small 3	G2	G3	G4	G5	G6	G1
Dataset 7: Position Medium 1	G1	G2	G3	G4	G5	G6
Dataset 8: Position Medium 1	G6	G1	G2	G3	G4	G5
...						
Dataset 35: Number Large 3	G3	G4	G5	G6	G1	G2
Dataset 36: Number Large 3	G2	G3	G4	G5	G6	G1

599 4.1 Experiment 1: Spotting the Difference

600 To evaluate how well our approach can assist observing changes between juxtaposed categorical scatterplots, we
 601 conduct an online “spot-the-difference” experiment through Amazon Mechanical Turk (AMT) with 136 participants.
 602

603 **Hypotheses.** We hypothesized that our approach would generally be more effective than the benchmark methods on
 604 the juxtaposed comparison tasks, and that this effect would vary based on *change magnitude* or *change type*.
 605

606 **H1.** Our color generation method (*C3-Palette Generation*) outperforms the benchmark conditions (*Random Assignment*,
 607 *Optimized Assignment*, *Alpha Blending* and *Palettailor*) on the task performance.
 608

609 **H2.** Our color assignment method (*C3-Palette Assignment*) using a color palette with a large range of brightness and
 610 saturation (*Tableau-20*) outperforms the benchmark conditions (*Random Assignment*, *Optimized Assignment*,
 611 *Alpha Blending* and *Palettailor*) on the task performance.
 612

613 **H3.** Other independent variables(*change type* and *change magnitude*) would also affect user performance on the task
 614 performance.
 615

616 **H4.** There would be an interaction effect between colorization methods and other independent variables(*change type*
 617 and *change magnitude*). Specifically, the difference between the effect of our methods (*C3-Palette Generation* and
 618 *C3-Palette Assignment*) and that of the benchmark methods (*Random Assignment*, *Optimized Assignment*, *Alpha*
 619 *Blending* and *Palettailor*) would change based on the different variable.
 620

621 4.1.1 Experimental Design.

622 **Task & Measures.** In this experiment, each participant was asked to perform a *spot-the-difference* task. Inspired by
 623

625 the Spot the Difference game where one needs to compare a pair of similar pictures to detect their differences [12], we
 626 asked participants to identify all the classes that have been changed in two scatterplots. At the beginning of each trial,
 627 the number of changed classes was provided. Each participant was asked to select all the changed classes by clicking
 628 the points belonging to these classes in either of the scatterplots.
 629

630 For each participant, we measured the *time* taken for each trial, and counted the errors (0/1) indicating whether
 631 the actual changed classes are aligned with the participant's response. Note that if any of the changed classes was
 632 mistakenly identified, the trial would be considered as "wrong" (1).
 633

634 While the participant was instructed to do the task "*as accurately as possible*", we set a 60-second time limit for
 635 each trial for fear that user might spend too much time on the trial. If the participant could not find all the changed
 636 classes during the time limit, they were directed to the next trial. There also will appear a "*Can't Find it*" button after 30
 637 seconds. This was done since we observed from the pilot study that when participants spent too much time on a single
 638 trial, they may decide to quit by selecting a class randomly(which will lead to an incorrect answer) or to spend more
 639 time till they get the correct answer or the time limit (which will lead to increasing time spent on the trial). This subject
 640 decision would add noise to our measurements. Thus we added a 30-second time limit, which was informed by our
 641 pilot study, where over 85% correct trials were completed within 30 seconds.
 642

643 **Experiment Organization.** We tested the effects of the 6 method conditions across 36 paired multi-class scatterplot
 644 datasets using a *between-subject* experiment design. To avoid ordering effects, where the participant would get familiar
 645 with a dataset after seeing it several times, each participant was assigned to a group and saw a specific subset of datasets
 646 under different conditions. We used a Latin Square grouping (see Table. 1) to organize the trials for each participant.
 647

648 In addition, some participants might apply a "shortcut" strategy when seeing a class that is obviously more salient
 649 than the others, especially under the *C3-Palette Assignment* and *C3-Palette Generation* conditions. Thus, for quality
 650 control, we added 4 sentinels which were very simple trials with only one changed class and a large change magnitude,
 651 and we assigned a de-saturated color to the changed class that made it less salient. We add these 4 distractor trials to
 652 each group to identify whether the participants is doing the task seriously and reject the results with more than two
 653 wrong trials.
 654

655 Finally, there were 6 participant groups and each of them had 40 trials in total. To further avoid learning effects
 656 between trials, we randomly shuffled the display orders of all scatterplot pairs, and randomly placed the two scatterplots
 657 in each pair on the left or right side.
 658

659 Table 2. Participants details for each task.
 660

661 662 663 664 Task & Group	665 666 667 668 669 670 Spotting the Difference		671 672 673 674 675 676 Counting class number	
	Pilot(28)	Formal(108)	Pilot(29)	formal(52)
Group 1	5	18	5	9
Group 2	5	17	5	8
Group 3	5	19	4	8
Group 4	3	17	5	9
Group 5	5	19	5	9
Group 6	5	18	5	9

672 **Pilot Study & Power Analysis.** We conducted a pilot study involving 28 participants to check the experimental setup
 673 and determine the parameters, such as the time limit for a trial. The results are shown in Fig. 5(a). Harnessing by the
 674 pilot study, we also obtained our expected effect sizes, which were in further fed into a power analysis. With an effect
 675 size of 0.5, the power of the study was approximately 0.8, which is acceptable for a pilot study.
 676

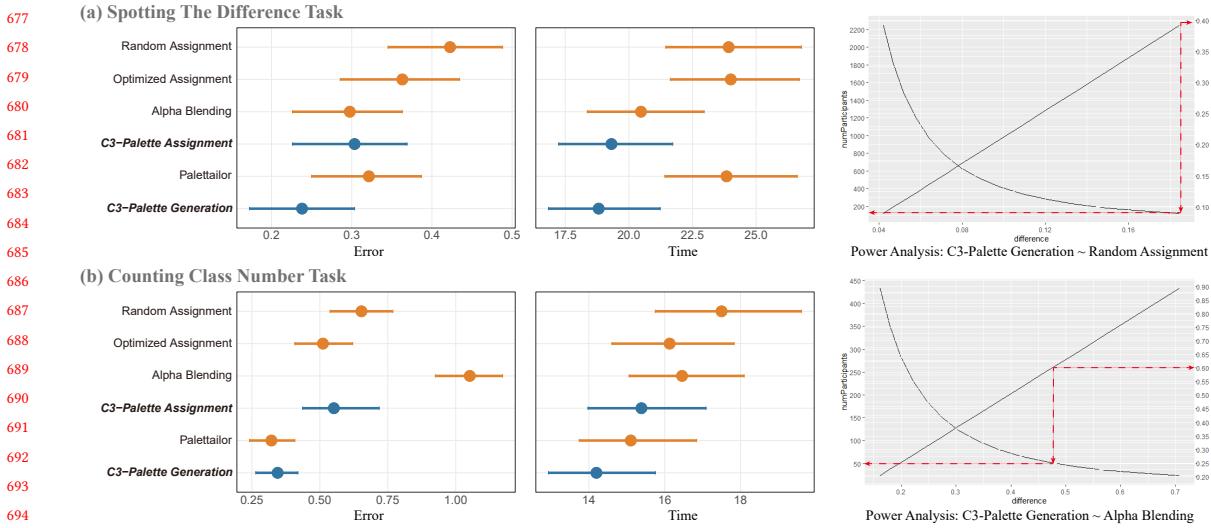


Fig. 5. Confidence interval plots and power analysis for the two pilot studies.

size Cohen's d of 0.4, alpha level of 0.05 and beta level of 0.8, the power analysis suggested a minimum number of 100 participants for the spot-the-difference task.

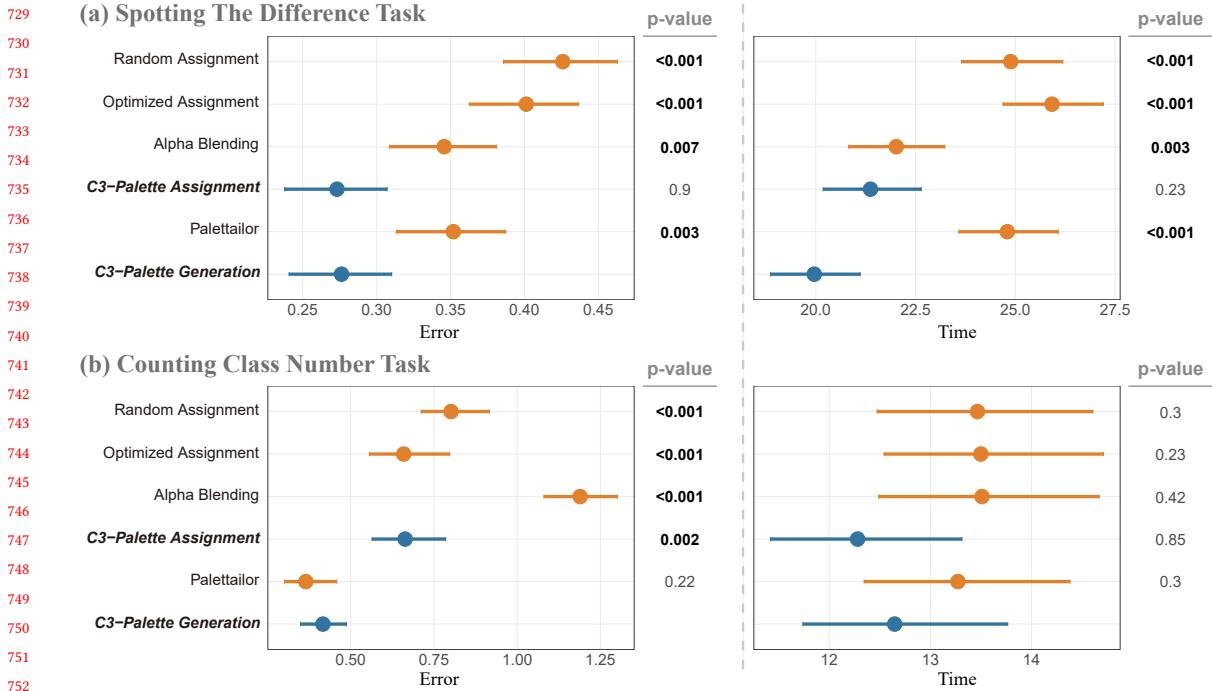
Participants. We recruited 108 participants(as shown in Table. 2) for the experiment on Amazon Mechanical Turk. According to the completion time in the pilot study, we paid each participant \$1.5 for the task based on the US minimum hourly wage. No participant claimed color vision deficiency on their informed consent.

Procedure. Each participant went through the following steps in our experiment: (i) viewing a user guide of the task and completing three training trials; (ii) completing each trial as accurately as possible; (iii) providing demographic information.

4.1.2 Results.

Following previous studies, we analyzed the results using 95% confidence intervals, and also conducted Mann-Whitney tests to compare the differences between conditions. The non-parametric test was used due to observations of non-normally distributed data from our pilot study. In addition, we computed the effect size using Cohen's d , i.e., the difference in means of the conditions divided by the pooled standard deviation. We used ANOVA to examine the interaction effect between variables.

Results of the online experiment are shown in Fig.6 (a). First, we found that our approach(*C3-Palette Assignment* and *C3-Palette Generation*) leads to a significantly lower error rate than all benchmark conditions. For consuming time, *C3-Palette Generation* has significantly less time ($p = 0.003$) than *Alpha Blending* condition while *C3-Palette Assignment* has no significant difference ($p = 0.095$), and our approach has significantly less time than all other benchmark conditions($p < 0.001$). The result indicates that our palette generation method(*C3-Palette Generation*) has a better performance than benchmark conditions in the “spot-the-difference” task (**H1 confirmed**). As for color palette with a larger range of brightness and saturation, our approach(*C3-Palette Assignment*) is better than most conditions and is at least comparable to *Alpha Blending* condition(**H2 confirmed**).



753 Fig. 6. Confidence interval plots and statistical tables for the two online controlled experiments. Error bars represent 95% confidence
754 intervals. Each table shows the statistical test results of C3-Palette Generation condition with other conditions, including the mean
755 with 95% confidence interval ($\mu \sim 95\%CI$), the W-value and p-value from the Mann-Whitney test, and the effect size ($d \sim 95\%CI$).
756

757 Second, we compared error and time with regard to different change magnitudes, and found that smaller magnitude
758 leads to larger error rate and consuming time (as shown in Fig.7 (a) left). This indicates that there exists an significant
759 interaction effect between *change magnitude* and performance, i.e., *change magnitude* would affect user performance.
760 We did the same test to *change type*, the results show that *point number change* is much more difficult than *point position
761 change*(H3 confirmed).

762 Finally, we did not find significant interaction effect between *colorization methods* and *change magnitude* or *change
763 type*, meaning that the effect of our method is not necessarily influenced by the magnitude of change between the two
764 scatterplots or the different change type of classes (H4 not confirmed).

765 4.2 Experiment 2: Counting Class Number

766 To evaluate whether our approach can fundamentally support the visual separability of the classes in each scatterplot, we
767 conduct an online “counting class number” experiment through Amazon Mechanical Turk (AMT) with 81 participants.
768 The experimental design was similar to the first study, but we set up with different task during the experiment. We
769 expected to see different patterns of the discriminability across different conditions. Specifically, our methods would
770 lead to a shorter error and time than *Random Assignment* and *Alpha Blending* conditions.

771 **Hypotheses.** We hypothesized that our approach would generally be more effective than the benchmark methods on
772 the discrimination tasks, and that this effect would not vary based on *change magnitude* or *change type*.
773

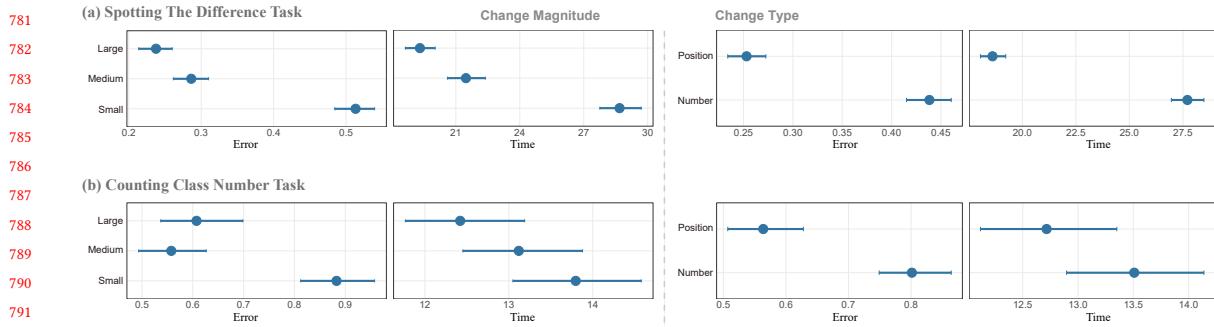


Fig. 7. Confidence interval plots for the two online controlled experiments. (left) Plots for *change magnitude* based on error and time; (right) plots for *change type* based on error and time.

- H1.** Our color generation method (*C3-Palette Generation*) outperforms the benchmark conditions (*Random Assignment*, *Optimized Assignment*, *Alpha Blending*) and our assignment method(*C3-Palette Assignment*), while is comparable to *Palettaior* on the task performance.
- H2.** Our color assignment method (*C3-Palette Assignment*) based on *Tableau-20* outperforms the benchmark conditions (*Random Assignment*, *Alpha Blending*), while is comparable to *Optimized Assignment* condition on the task performance.
- H3.** Other independent variables(*change magnitude* and *change type*) would have no effect on discrimination task between different conditions.
- H4.** There would be no interaction effect between colorization methods and other independent variables(*change type* and *change magnitude*).

4.2.1 Experimental Design.

Task & Measures. Following previous methodologies [29, 41], each participant was asked to perform a *counting class number* task. We asked participants to identify how many classes(colors) are there in the given two scatterplots and then choose an answer among several options below the two scatterplots. We recorded the participant's answer and response time for each trial, and counted the *error* by calculating the differences between the participant's answer and the actual number of classes(each scatterplot has 8 classes in our experiment).

Pilot Study & Power Analysis. This setting is similar to Experiment 1. We invited 29 participants to do the pilot study and the results were in further fed into a power analysis. With an effect size Cohen's d of 0.6, the power analysis suggested a minimum number of 50 participants for the discriminability task. See the supplementary material for more details.

Participants. We finally recruited 52 participants(as shown in Table. 2) for the experiment on Amazon Mechanical Turk. According to the completion time in the pilot study, we paid each participant \$1.5 for the task based on the US minimum hourly wage. No participant claimed color vision deficiency on their informed consent.

4.2.2 Results.

Results of this visual separability experiment are shown in Fig.6 (b). Through this study we found that first *C3-Palette Generation* is comparable to *Palettaior* while leads to a significantly lower error rate($p \leq 0.001$) than all other

benchmark conditions. Specifically, *C3-Palette Generation* has a significantly lower error rate($p = 0.002$) than *C3-Palette Assignment*(**H1** confirmed). Second, *C3-Palette Assignment* has higher performance than the benchmark conditions (*Random Assignment*, *Alpha Blending*) and is comparable to *Optimized Assignment*(**H2** confirmed). For other independent variables, as shown in Fig.7 (b), we found that there existed a significant difference between *Small change magnitude* and *Medium* and *Large*. *Point position change* has a much lower error rate than *point number change*. And their time has both a tendency to gradually increase. This indicates that *change magnitude* and *change type* might have an effect on discrimination task between different conditions (**H3** not confirmed). Finally, we did not find significant interaction effect between *colorization methods* and *change magnitude* or *change type*, meaning that the effect of different methods for visual discriminability is not necessarily influenced by the magnitude of change between the two scatterplots or the different change type of classes (**H4** confirmed).

4.3 Discussion

In summary, we evaluated the effectiveness of our approach against the benchmark conditions through two online studies. We found that first, our methods outperform the benchmark methods on juxtaposed comparison tasks, and their effects are not necessarily influenced by the change magnitude of the two scatterplots or the change type of each class. The performance of *Optimized Assignment* is comparable to *Random Assignment*, this is reasonable, since *Optimized Assignment* mainly cares about the visual separability of different classes, thus it might assign the less salient color to the changed class while *Random Assignment* would assign salient color even though the whole separability of the scatterplot is not very good. This also provides an explanation for *Alpha Blending* which is based on the result of *Optimized Assignment*. Second, our experimental methods (*C3-Palette Generation* and *C3-Palette Assignment*) generally support the fundamental visual separability of the classes. It is worth noting that the error rate of *C3-Palette Generation* is comparable to *Palettailor* which is the start-of-the-art palette generation method for visual discriminability, while *C3-Palette Assignment* is comparable to *Optimized Assignment* which is the start-of-the-art palette assignment method for visual discriminability. This indicates that our approach maintains the class distinction of the scatterplot while enhances the class saliency to help observe changes between different scatterplots. Third, we found that *change magnitude* and *change type* influence the performance of the *counting class number* task. The potential explanation is that large change between scatterplots will attract participants' attention, thus make it easy to distinct different classes. This is also reasonable for *change type* since point position change is easier to distinguish than point number change. It's obvious that *Alpha Blending* has a much lower error rate than other methods for discrimination task. As one of the participants said, "The ones that were harder were ones that had colors that when they overlapped would change color. It made it hard to tell if it was the same color or if it was a new color. When the colors were uniform and all the same opacity, it was much easier." *Alpha Blending* condition changes the opacity of unchanged classes to make the unchanged classes more distinct, but this will generate new color from color blending, so as to make it hard to distinct colors.

Some limitations exist in our evaluation. First, our experiment mainly focuses on error rate and time consuming, while other measurements are not explored, such as click order of the changed classes and time consuming for each click. These might reflect some interesting results for different *cluster type*. Second, our experiment focuses on identifying the differences between two scatterplots, which is a simplified situation, since in real-world cases often more than two visualizations are compared. Third, we cannot further analyze the effect of *change type*, given the current study design, though we did observe some trends that for certain types of change, our methods are more effective. That brings us to a series of more fundamental questions: how can we properly define the types of changes? What is the just noticeable

885 change magnitude for each change type? Further research is needed to answer these questions so that our approach
 886 can be thoroughly evaluated.
 887

888 4.4 Experiment 3: Finding Max Delta

890 In addition to scatterplots, we also test the performance of our method for bar charts through the *MAXDELTA* task
 891 which is used in Ondov et al.'s [34] comparison task. The setup of this task is very similar to Ondov et al. [34], thus we
 892 mainly describe the difference between the two below.
 893

894 4.4.1 Experimental Design.

895 **Task & Measure.** *MAXDELTA task.* Following the methodology by Ondov et al. [34], we asked participants to find the
 896 bar that had the biggest difference in the two charts. The titer value, i.e., the distractor delta which is used to control the
 897 largest difference between the two bar charts, is recorded to measure the precision degree for people to make judgments
 898 about adjacent bar charts. Larger titer value indicates a higher performance.
 899

900 **Conditions & sample palette.** Similar to the scatterplot experiment, we also tested the six conditions, including two
 901 experimental methods(*C3-Palette Assignment* and *C3-Palette Generation*), as well as four benchmark conditions(*Random*
 902 *Assignment*, *Optimized Assignment*, *Alpha Blending* and *Palettaior*). We used Tableau 20 palette for the assignment
 903 method due to its large range of brightness and saturation. Specifically, the original *Optimized Assignment* [41] did not
 904 apply their method to bar chart, we extended it by treating each bar as a point. *Alpha Blending* is based on the result
 905 of *Optimized Assignment* by changing the opacity of each class. Different to the scatterplot experiment, we run these
 906 methods in real-time since the bar charts are generated dynamically.
 907

908 **Bar chart generation.** All the stimulus datasets used in this experiment are generated by the titer staircase method [34]
 909 in real time. Each pair of bar charts are generated differently depending on participant performance in the previous
 910 trial. For each condition method, the first trial generate bar charts with a titer of 0.5. A correct response made the titer
 911 of the next trial increase with +0.3 while an erroneous answer lead to a decrease of -0.1. To prevent participants from
 912 too many trials with low difficulty, we set 0.75 as the largest titer value. Each bar chart consists of 7 data points same as
 913 Ondov et al. [34].
 914

915 **Procedure.** The procedure of this experiment is same with [34] except
 916

917 5 INTERACTIVE SYSTEM

918 To help users interactively design colors for comparing multi-class scatterplots, we developed a web-based multi-view
 919 visualization tool ². The screenshot of the interface can be found in the supplemental material. This tool consists of four
 920 coordinated views: (a) a settings panel, (b) a control panel for adjusting importance threshold κ and even importance
 921 value of each class, (c) the juxtaposed visualizations, and (d) a history view. The control panel shows the decision which
 922 classes are highlighted, and the history view allows to quickly explore and access previous color mappings.
 923

924 After uploading multiple categorical scatterplots, the user can either choose a default color palette or use our system
 925 to automatically generate color palettes. In this case, the system automatically finds an optimal color mapping scheme
 926 to colorize the input data, while each class is encoded as a circle where the x-axis represents class label and the y-axis
 927 indicates the importance of each class. By default, the importance is represented by the change degree and κ is set to
 928 zero. User can drag the circle to modify the corresponding importance value. The κ is controlled by a black circle on the
 929 y-axis which can also be dragged to modify. Our system finds a color mapping scheme to highlight the classes with
 930

931 ²<https://c3-palette.github.io/>

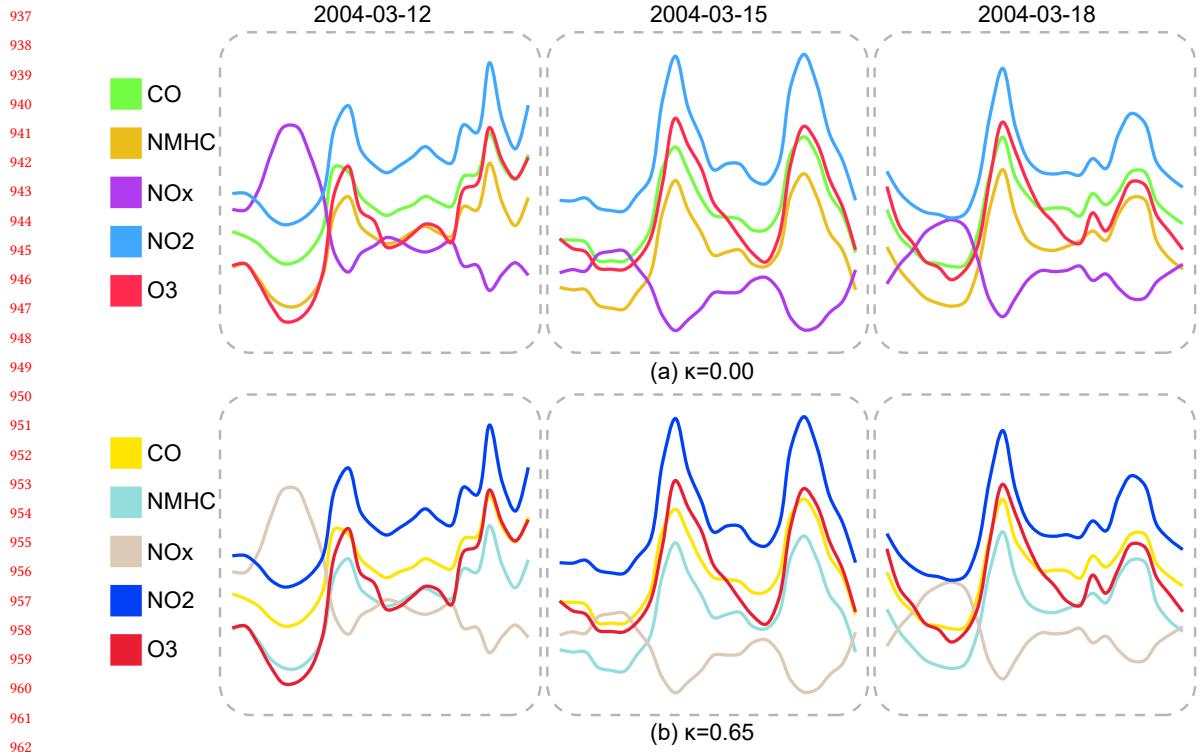


Fig. 8. Exploring the changes of gases in an air quality data set [9]. (a) The automatic generated palette creates salient colors for all lines. (b) Two selected lines are popping out from the other lines in three views.

large importance and renders the classes in ascending order of the corresponding importance. If users like the color mapping scheme, they can save it to the history view.

5.1 Case Study

To shed further light onto the ecological validity of our approach, we conducted a case study on a real-world categorical dataset visualized with line charts. Here, we analyze an air quality data set [9] that contains hourly responses of a gas multi-sensor device deployed in an Italian city from March 12 to March 18, 2014. Fig. 8 shows the juxtaposed line graphs encoded by our generated color palette, where each gas type is represented by a line with a unique color. We can see that all gases are encoded with highly salient colors, making it hard to explore changes of specific gases. This is reasonable because the default κ is zero, but all gases have large changes. Thanks to our interaction mechanism, users can directly select classes of interest to be highlighted, the $f(\theta)$ values of the other classes are then automatically set to -1. Using the hue-preserving palette generation method, the produced palette lets the selected lines pop out from the others (see Fig. 8(b)). Hence, users can easily explore the changes of the selected two gases (NO₂ and O₃) in the three juxtaposed views.

989 6 DISCUSSION AND FUTURE WORK

990 Our work concentrated on horizontally juxtaposed comparisons to detect changes between multiple datasets. Although
 991 detecting changes is a fundamental visual comparison task, its optimal color palette might not be appropriate for
 992 understanding other analytical comparison tasks (e.g., the correlation tasks [34]. Future work needs to investigate the
 993 effectiveness and extensions of our approach for such comparison tasks. Furthermore, our approach produces colors
 994 with salient hue to highlight classes with large changes, but those colors do not visually indicate the ranking of class
 995 changes. It would be helpful to associate the color ordering constraint [5] with the degree of changes, so that the
 996 ranking of class changes can be shown clearly.
 997

998 Second, our study only evaluated the effectiveness of our palettes with horizontal juxtaposed charts, while there are
 999 a few different layout methods such as vertical arrangement, mirrored arrangement, overlaid, and animation. Previous
 1000 studies [34] show that animation performs well in identifying the largest difference and we will conduct studies to
 1001 learn how well our palette works in this setting.
 1002

1003 Third, plotting difference between two compared data series is a widely used method [14]. This method might work
 1004 well for bar and line charts, but the difference between two scatterplots is hard to be visualized in a meaningful way.
 1005 Plotting multiple classes with facets [43] can bypass this issue, but it requires large space when the number of classes is
 1006 large. We will study and compare the effectiveness of these methods with different sized data.
 1007

1008 Last, while we only studied the interaction effect between change magnitude and different colorization methods, we
 1009 plan to investigate how this effect is influenced by different types of changes, such as point number, center position
 1010 and shape. The order of rendering is critical for comparison task and we treat it simply in this paper by rendering
 1011 less important classes first. But when there are multiple important large classes at same positions, the less important
 1012 class might be overlapped and hard to distinct. Thus a professional render order algorithm is necessary for multi-class
 1013 scatterplot rendering.
 1014

1015 REFERENCES

- [1] D. Albers, C. Dewey, and M. Gleicher. 2011. Sequence Surveyor: leveraging Overview for Scalable Genomic Alignment Visualization. *IEEE Transactions on Visualization and Computer Graphics* 17, 12 (2011), 2392–2401. <https://doi.org/10.1109/TVCG.2011.232>
- [2] M. Aupetit and M. Sedlmair. 2016. SepMe: 2002 New visual separation measures. In *2016 IEEE Pacific Visualization Symposium*. 1–8. <https://doi.org/10.1109/PACIFICVIS.2016.7465244>
- [3] Patrick Baudisch and Carl Gutwin. 2004. Multiblending: displaying overlapping windows simultaneously without the drawbacks of alpha blending. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 367–374.
- [4] Ali Borji, Ming-Ming Cheng, Qibin Hou, Huaiyu Jiang, and Jia Li. 2019. Salient object detection: a survey. *Computational Visual Media* 5, 2 (2019), 117–150. <https://doi.org/10.1007/s41095-019-0149-9>
- [5] R. Bujack, T. L. Turton, F. Samsel, C. Ware, D. H. Rogers, and J. Ahrens. 2018. The Good, the Bad, and the Ugly: a Theoretical Framework for the Assessment of Continuous Colormaps. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 923–933. <https://doi.org/10.1109/TVCG.2017.2743978>
- [6] H. Chen, W. Chen, H. Mei, Z. Liu, K. Zhou, W. Chen, W. Gu, and K. Ma. 2014. Visual Abstraction and Exploration of Multi-class Scatterplots. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (2014), 1683–1692. <https://doi.org/10.1109/TVCG.2014.2346594>
- [7] J. Chuang, D. Weiskopf, and Torsten Möller. 2009. Energy Aware Color Sets. *Computer Graphics Forum* 28 (2009). <https://doi.org/10.1111/j.1467-8659.2009.01359.x>
- [8] Charles E. Connor, Howard E. Egeth, and Steven Yantis. 2004. Visual Attention: bottom-Up Versus Top-Down. *Current Biology* 14, 19 (2004), R850–R852. <https://doi.org/10.1016/j.cub.2004.09.041>
- [9] S. De Vito, E. Massera, M. Piga, L. Martinotto, and G. Di Francia. 2008. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sensors and Actuators B: Chemical* 129, 2 (2008), 750–757. <https://doi.org/10.1016/j.snb.2007.09.060>
- [10] James T Enns. 1990. Three-dimensional features that pop out in visual search. (1990).
- [11] H. Fu, X. Cao, and Z. Tu. 2013. Cluster-Based Co-Saliency Detection. *IEEE Transactions on Image Processing* 22, 10 (2013), 3766–3778. <https://doi.org/10.1109/TIP.2013.2260166>

- [12] Eiji Fukuba, Hajime Kitagaki, Akihiko Wada, Kouji Uchida, Shinji Hara, Takafumi Hayashi, Kazushige Oda, and Nobue Uchida. 2009. Brain Activation during the Spot the Differences Game. *Magnetic Resonance in Medical Sciences* 8, 1 (2009), 23–32. <https://doi.org/10.2463/mrms.8.23>
- [13] M. Gleicher. 2018. Considerations for Visualizing Comparison. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 413–423. <https://doi.org/10.1109/TVCG.2017.2744199>
- [14] Michael Gleicher, Danielle Albers, Rick Walker, Ilir Jusufi, Charles D. Hansen, and Jonathan C. Roberts. 2011. Visual Comparison for Information Visualization. *Information Visualization* 10, 4 (2011), 289–309. <https://doi.org/10.1177/1473871611416549>
- [15] C. C. Gramazio, D. H. Laidlaw, and K. B. Schloss. 2017. Colorgorical: creating discriminable and preferable color palettes for information visualization. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 521–530. <https://doi.org/10.1109/TVCG.2016.2598918>
- [16] Mark Harrower and Cynthia A. Brewer. 2003. ColorBrewer.org: an online tool for selecting colour schemes for maps. *The Cartographic Journal* 40, 1 (2003), 27–37. <https://doi.org/10.1179/000870403235002042>
- [17] Christopher G Healey, Kellogg S Booth, and James T Enns. 1995. Visualizing real-time multivariate data using preattentive processing. *ACM Transactions on Modeling and Computer Simulation* 5, 3 (1995), 190–221. <https://doi.org/10.1145/217853.217855>
- [18] Christophe Hurter, Mathieu Serrurier, Roland Alonso, Gilles Tabart, and Jean-Luc Vinot. 2010. An Automatic Generation of Schematic Maps to Display Flight Routes for Air Traffic Controllers: structure and Color Optimization. In *Proceedings of the International Conference on Advanced Visual Interfaces*. 233–240. <https://doi.org/10.1145/1842993.1843034>
- [19] L. Itti, C. Koch, and E. Niebur. 1998. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20, 11 (1998), 1254–1259. <https://doi.org/10.1109/34.730558>
- [20] David E. Jacobs, Dan B. Goldman, and Eli Shechtman. 2010. Cosaliency: where People Look When Comparing Images. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology*. 219–228. <https://doi.org/10.1145/1866029.1866066>
- [21] H. Jänicke and M. Chen. 2010. A Salience-based Quality Metric for Visualization. *Computer Graphics Forum* 29, 3 (2010), 1183–1192. <https://doi.org/10.1111/j.1467-8659.2009.01667.x>
- [22] N. Jardine, B. D. Ondov, N. Elmquist, and S. Franconeri. 2020. The Perceptual Proxies of Visual Comparison. *IEEE Transactions on Visualization and Computer Graphics* 26, 1 (2020), 1012–1021. <https://doi.org/10.1109/TVCG.2019.2934786>
- [23] Hye-Rin Kim, Min-Joon Yoo, Henry Kang, and In-Kwon Lee. 2014. Perceptually-Based Color Assignment. *Computer Graphics Forum* 33, 7 (2014), 309–318. <https://doi.org/10.1111/cgf.12499>
- [24] Y. Kim and A. Varshney. 2006. Saliency-guided Enhancement for Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics* 12, 5 (2006), 925–932. <https://doi.org/10.1109/TVCG.2006.174>
- [25] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 2, 1-2 (1955), 83–97. https://doi.org/10.1007/978-3-540-68279-0_2
- [26] S. Lee, M. Sips, and H. Seidel. 2013. Perceptually Driven Visibility Optimization for Categorical Data Visualization. *IEEE Transactions on Visualization and Computer Graphics* 19, 10 (2013), 1746–1757. <https://doi.org/10.1109/TVCG.2012.315>
- [27] Sharon Lin, Julie Fortuna, Chinmay Kulkarni, Maureen Stone, and Jeffrey Heer. 2013. Selecting Semantically-Resonant Colors for Data Visualization. *Computer Graphics Forum* 32, 3 (2013), 401–410. <https://doi.org/10.1111/cgf.12127>
- [28] María-Jesús Lobo, Emmanuel Pietriga, and Caroline Appert. 2015. An Evaluation of Interactive Map Comparison Techniques. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 3573–3582. <https://doi.org/10.1145/2702123.2702130>
- [29] K. Lu, M. Feng, X. Chen, M. Sedlmaier, O. Deussen, D. Lischinski, Z. Cheng, and Y. Wang. 2021. Palettaior: discriminable colorization for categorical data. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2021), 475–484. <https://doi.org/10.1109/TVCG.2020.3030406>
- [30] S. LYi, J. Jo, and J. Seo. 2021. Comparative Layouts Revisited: design Space, Guidelines, and Future Directions. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2021), 1525–1535. <https://doi.org/10.1109/TVCG.2020.3030419>
- [31] G. M. Machado, M. M. Oliveira, and L. A. F. Fernandes. 2009. A Physiologically-based Model for Simulation of Color Vision Deficiency. *IEEE Transactions on Visualization and Computer Graphics* 15, 6 (2009), 1291–1298. <https://doi.org/10.1109/TVCG.2009.113>
- [32] L. E. Matzen, M. J. Haass, K. M. Divis, Z. Wang, and A. T. Wilson. 2018. Data Visualization Saliency Model: a Tool for Evaluating Abstract Data Visualizations. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 563–573. <https://doi.org/10.1109/TVCG.2017.2743939>
- [33] Tamara Munzner, François Guimbretière, Serdar Tasiran, Li Zhang, and Yunhong Zhou. 2003. TreeJuxtaposer: scalable Tree Comparison Using Focus+Context with Guarantees of Visibility. *ACM Transactions on Graphics* 22, 3 (2003), 453–462. <https://doi.org/10.1145/882262.882291>
- [34] B. Ondov, N. Jardine, N. Elmquist, and S. Franconeri. 2019. Face to face: evaluating visual comparison. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 861–871. <https://doi.org/10.1109/TVCG.2018.2864884>
- [35] Zening Qu and Jessica Hullman. 2017. Keeping multiple views consistent: constraints, validations, and exceptions in visualization authoring. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 468–477. <https://doi.org/10.1109/TVCG.2017.2744198>
- [36] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. 2000. The Earth Mover's Distance as a Metric for Image Retrieval. *International Journal of Computer Vision* 40, 2 (2000), 99–121. <https://doi.org/10.1023/A:1026543900054>
- [37] V. Setlur and M. C. Stone. 2016. A Linguistic Approach to Categorical Color Assignment for Data Visualization. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (2016), 698–707. <https://doi.org/10.1109/TVCG.2015.2467471>
- [38] Tableau Software. [n.d.]. The tableau visualization system. <http://www.tableausoftware.com/>.
- [39] C. Tominski, C. Forsell, and J. Johansson. 2012. Interaction Support for Visual Comparison Inspired by Natural Behavior. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2719–2728. <https://doi.org/10.1109/TVCG.2012.237>

- 1093 [40] C. Tominski, G. Fuchs, and H. Schumann. 2008. Task-driven color coding. In *Proceedings of 12th International Conference Information Visualisation*.
1094 373–380. <https://doi.org/10.1109/IV.2008.24>
- 1095 [41] Yunhai Wang, Xin Chen, Tong Ge, Chen Bao, Michael Sedlmair, Chi-Wing Fu, Oliver Deussen, and Baoquan Chen. 2019. Optimizing color assignment
1096 for perception of class separability in multiclass scatterplots. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 820–829.
1097 <https://doi.org/10.1109/TVCG.2018.2864912>
- 1098 [42] Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. 2000. Guidelines for Using Multiple Views in Information Visualization. In
1099 *Proceedings of the Working Conference on Advanced Visual Interfaces*. 110–119. <https://doi.org/10.1145/345513.345271>
- 1100 [43] Hadley Wickham et al. 2009. Elegant graphics for data analysis. *Media* 35, 211 (2009), 10–1007.
- 1101 [44] Dingwen Zhang, Huazhu Fu, Junwei Han, Ali Borji, and Xuelong Li. 2018. A review of co-saliency detection algorithms: fundamentals, applications,
1102 and challenges. *ACM Transactions on Intelligent Systems and Technology* 9, 4 (2018), 1–31. <https://doi.org/10.1145/3158674>
- 1103 [45] L. Zhou and C. D. Hansen. 2016. A Survey of Colormaps in Visualization. *IEEE Transactions on Visualization and Computer Graphics* 22, 8 (2016),
1104 2051–2069. <https://doi.org/10.1109/TVCG.2015.2489649>
- 1105
- 1106
- 1107
- 1108
- 1109
- 1110
- 1111
- 1112
- 1113
- 1114
- 1115
- 1116
- 1117
- 1118
- 1119
- 1120
- 1121
- 1122
- 1123
- 1124
- 1125
- 1126
- 1127
- 1128
- 1129
- 1130
- 1131
- 1132
- 1133
- 1134
- 1135
- 1136
- 1137
- 1138
- 1139
- 1140
- 1141
- 1142
- 1143
- 1144