$\mathbb{C}^3$-palette: Co-saliency based Colorization for Comparing Multi-class Scatterplots

KECHENG LU, Shandong University, China
MI FENG, Twitter Inc., USA
YUNHAI WANG, Shandong University, China

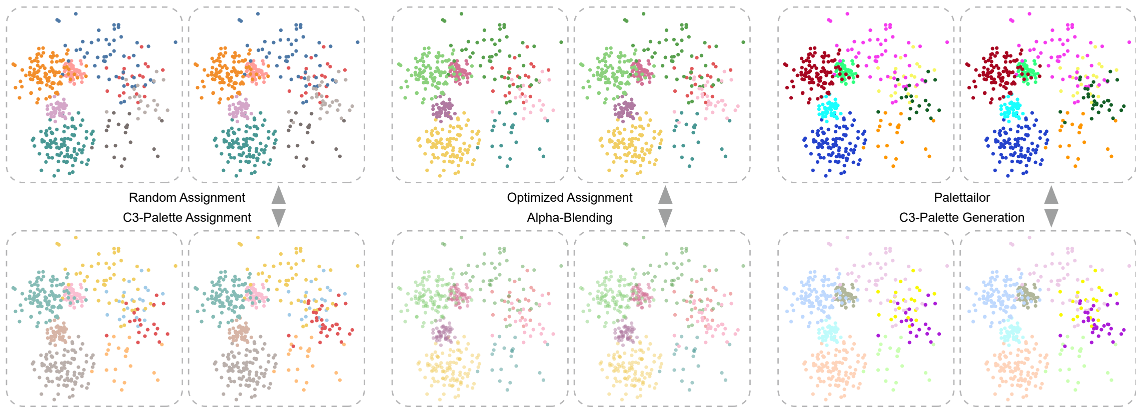


Fig. 1. Results for different conditions of two categorical scatterplots comparison: (left top) Random Assignment; (left bottom) C3-Palette Assignment; (center top) Optimized Assignment [41]; (center bottom) Applying Alpha-Blending on Optimized Assignment, all the classes' alpha are set to 0.5 except the changed class; (right top) Palettailor [29]; (right bottom) C3-Palette Generation. Our system unifies the palette assignment and palette generation to single or multiple scatterplots in a data-aware manner.

Visual comparison within juxtaposed views is an essential part of interactive data analysis. In this paper, we propose a co-saliency model to characterize the most co-salient features among juxtaposed labeled data visualizations while maintaining class discrimination in individual visualizations. Based on this model, we present a comparison-driven color design framework, enabling automatic selection and generation of colors that maximizes co-saliency among juxtaposed visualizations. We conduct a numeric study, an online controlled experiment and a lab study with eye tracking to compare our colorizations with results produced by existing single view-based color design methods. We further present an interactive system and conduct one case study to demonstrate our usefulness for comparisons of juxtaposed line charts. The results show that our approach is able to generate high quality color palettes in support of visual comparison of juxtaposed categorical visualizations.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

Additional Key Words and Phrases: datasets, neural networks, gaze detection, text tagging

Manuscript submitted to ACM

ACM Reference Format:
Kecheng Lu, Mi Feng, and Yunhai Wang. 2018. $\mathbb{C}^3$-palette: Co-saliency based Colorization for Comparing Multi-class Scatterplots. In *Woodstock '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

1 INTRODUCTION

Comparison is an indispensable task in data analysis and visualization. It often involves searching for categories (classes) with large or small changes among multiple categorical datasets. Such comparison is usually achieved through juxtaposition of multiple visualizations [13, 30] such as multi-class scatterplots, line and bar charts. Regardless of the visualization type, each class is commonly encoded by a unique color. While color plays an important role in helping viewers see differences between juxtaposed views [2, 13, 40], finding an appropriate color mapping scheme to ease the process for comparative visualization is a challenging and yet unexplored problem.

The most common way to colorize juxtaposed views is finding an appropriate color mapping for one artificially selected view while judging how well it fits to the other views. Such a trial and error procedure might converge to a desirable color mapping; however, its required efforts significantly increase with the numbers of classes and views. Although existing automated color selection approaches [6, 29, 41] can alleviate the effort for single view colorization, the obtained color mapping might not be able to clearly reveal similarities or differences among multiple views. For example, the optimized assignment[41] of the Tableau palette in Fig. 1(middle top) creates a visualization with better class separation than the one generated by random assignment in Fig. 1(left top), although the changed pink class is hard to be identified. As far as we know, few existing visualization-oriented color selection tools (e.g., ColorBrewer [16] or Palettailor [29]) allow for colorizing multi-view visualizations, let alone supporting comparisons in juxtaposed views.

There are two simple ways to assist comparison task, one is using alpha-blending to highlight concerned classes, the other is using faceting by groups with the groups highlighted on top of all the cases []. These two methods only cared about highlighting the concerned classes while make other classes invisible or hard to discriminate. However, user might want to explore one of the scatterplot rather than change the visualization, i.e., alpha-blending need to set all classes' opacity to 1.0 while facet need to show other classes in one visualization. As far as we know, there does not exist a method that unifies both highlighting important parts while maintaining good class separability for comparison task.

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

By integrating our co-saliency model into existing data-aware color assignment and categorical data colorization tools [29, 41], we can automatically select/generate color mappings that maximize co-saliency among juxtaposed visualizations. The resulted color mapping scheme makes the classes with large importance pop out from the context and attract viewers' attention, while maximizing the perceptual separability between classes in individual visualizations. By doing so, the major issue [39] of the juxtaposition is that humans have limited visual memory is greatly alleviated and the visual search can be done with less cognitive cost [17]. Fig. 1(left bottom) shows the results generated by performing co-saliency based color assignment, where the changed class in red is easier to be spot than the one in Fig. 1(middle bottom). The pre-attentive "pop out" effect of teh class is further enhanced in Fig. 1(right bottom) by using our colorization method.

Since our method is based on previous work [29, 41], we employ a carefully design for upward compatible. That is, our method can be used for both single or multiple scatterplots. Thus we provide a unified framework for scatterplot colorization, including color assignment with pre-defined palette and automatic palette generation. Our method can highlight interesting classes in single scatterplot due to the importance factor which can be manually adjusted by user while maintains the class separability.

Scatterplots are one of the most commonly used chart type for visualizing multi-class data, and it is harder to compare categorical scatterplots than line charts or bar charts, due to this, we mainly use them to evaluate our framework. For each of 36 multi-class scatterplots generated by using the method of Lu et al. [29], we produce its counterpart by changing properties (point number, point position) of several randomly selected classes. After scatterplot generation, we create the experiment data by applying different colorization method for each scatterplot pair, including two experimental methods based on our approach (*C3-Palette Assignment*, *C3-Palette Generation*) and four benchmark methods (*Random Assignment*, *Optimized Assignment*, *Alpha Blending* and *Palettailor*). With this dataset, we first conducted a pilot study to verify the validity of our experiment setting and then ran this user study to investigate how well our generated palettes help users to identify changed classes. Second, we conducted another pilot study for the validation of visual discriminability task and then ran this study to explore the high efficiency of our method for class separability. These two experiments are all executed through the Amazon Mechanical Turk (AMT) with 217 participants in total. Last, we conducted a case study to show how our system helps for juxtaposed comparison of multiple categorical scatterplots. The results show that our approach is able to produce color mapping optimized for supporting comparison and aligned with the state-of-the-art palettes in maximizing perceptual class separability.

We furthermore develop a web-based color design tool [1], using coordinated views for users to explore the relationship among multiple data with different color mapping schemes. The main contributions of this paper are as follows:

- We propose a multi-class data visualization co-saliency model for measuring the importance of each data item shown in juxtaposed visualizations and use this metric to automatically generate color mapping schemes for effective comparisons;
- We provide an interactive tool that show how our approach can be used for helping visual comparison of multiple categorical scatterplots or even highlighting important classes within single scatterplot; and
- We evaluate the effectiveness of the resulting color mapping schemes in supporting both visual comparison and visual discriminability with two online user studies and a case study (Section 5).

[1] https://c3-palette.github.io/

2 RELATED WORK

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

2.1 Visual Comparison

Visual comparison is an essential part of interactive data analysis, which is regarded as a high-level "compound task." Gleicher et al. [14] provide a systematic review of techniques developed for better supporting comparison and summarize three basic layout designs for comparative visualization, including *juxtaposition*, *superposition* and *explicit encoding*. Among them, juxtaposition places different datasets in different views without any change to the original visualization design and thus it is commonly used in many applications [2, 28, 33]. However, it often causes cognitive burden because users need to maintain a mental image of one view for comparing with another view [30]. Recently, Ondov et al. [34] and Jardine et al. [22] evaluated the perceptual effectiveness of different layouts for bar charts comparison with a few low-level tasks, which show that juxtaposition is less effective in some tasks like finding "biggest delta between items." Accordingly, Gleicher et al. [14] and L'Yi et al. [30] both suggested to carefully design visual encoding for improving its effectness. Our method facilitates visual comparison of categorical data by improving the visual search with the pop-out effect [9] induced by our proposed color mapping scheme.

2.2 Color Design

For a complete review of color design for visualization, we refer readers to survey papers [40, 44]. We limit our discussion to the techniques related to color design for categorical data visualization including color mapping optimization, color palette generation, and color design for multi-view visualization.

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

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

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

2.3 Visual Saliency & Co-saliency

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

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

Image Co-Saliency. Unlike single image based saliency model, the co-saliency model estimates the saliency (importance) of each pixel within the context of multiple related images. Jacobs et al. [20] developed the first co-saliency model for highlighting the most salient differences between two compared images. Later, this concept has been extended for discovering common and salient objects/foregrounds from image collections [43]. Inspired by the original model [20], our work attempts to design an appropriate color mapping for visualizing the most co-salient features among juxtaposed labeled data visualizations. Following their findings that the co-salient features can be effectively characterized by fusing image changes and single image contrast together, our co-saliency model relies on two factors: the class contrast in individual views and global features from in-between views (e.g., class structure changes).

3 CO-SALIENCY BASED COLOR DESIGN

Given multiple categorical scatterplots with the same class labels (or a subset thereof), each scatterplot $\mathbf{X}^j$ has M classes and n_j data items $\{\mathbf{x}^j_1, \cdots, \mathbf{x}^j_{n_j}\}$, where each $\mathbf{x}^j_t$ has a label $l(\mathbf{x}^j_t)$ and the i-th class (with n^j_i data points) consists of

$\{\mathbf{x}^j_{i,1}, \cdots, \mathbf{x}^j_{i,n^j_i}\}, i \in \{1, \cdots, m\}$. All visualizations use the same background color $\mathbf{c}_b$ and the same color mapping scheme $\tau : L \mapsto c$. Our goal is to find the best mapping τ that supports effective comparison of multiple categorical scatterplots.

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

(i) **DR1:** highlighting the most concerned classes between visualizations as much as possible for an efficient comparison;

(ii) **DR2:** maximizing the visual discrimination between classes in individual visualizations for an efficient exploration of multi-class data; and

(iii) **DR3:** providing flexible interactions for the exploration of relationships among the compared datasets.

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 most salient features in multiple visualizations. To address this issue, we propose a categorical visualization co-saliency model that calculates the saliency of each data item in the context of other similar visualizations. Integrating this model into the objective of state-of-the-art color mapping selection or generation frameworks [29, 41], we can generate proper color mappings to highlight salient differences between juxtaposed categorical visualizations.

3.1 Co-saliency for Multi-class Scatterplots

Following the definition of image co-saliency [20], we model the class co-saliency with two factors: class importance between scatterplots and class contrast within scatterplots. The class importance describes how much each class should stand out from the visualizatioin. While the class contrast measures the distinctness from neighboring classes and the background, which is similar to perceptual class separability [3, 41]. Hence, we define two types of class contrasts: contrast with neighboring classes and contrast to the background. Analogous to bottom-up image co-saliency models [10, 20], the co-saliency of the ith class is defined as the product between class importance and class contrast score to emphasize the target class, and the co-saliency for M classes:

$$E_{CoS} = \sum_i \left(\sum_j \frac{1}{n^j_i} (\lambda \alpha^j_i + (1-\lambda)\beta^j_i) \right) \exp(\theta_i) \tag{1}$$

where θ_i is the importance of the ith class, α^j_i is the contrast with neighboring classes of the ith class in the jth scatterplot, β^j_i is the contrast to the background, and λ is a weight between them. To better support DR1, we apply an exponential function to enlarge the weight of class importance, thus makes the target class easy to get a discriminable color from the optimization process.

Class Contrast. Given the jth scatterplot, we define the local class contrast with both point distinctness and point contrast with background [41] based on the neighbors calculated by α-Shape [29]. For each data point $\mathbf{x}^j_t$, we define its point distinctness as:

$$\gamma(\mathbf{x}^j_t) = \frac{1}{|\Omega^j_t|} \sum_{\mathbf{x}^j_p \in \Omega^j_t} \frac{\Delta\epsilon(\tau(l(\mathbf{x}^j_t)), \tau(l(\mathbf{x}^j_p)))}{d(\mathbf{x}^j_t, \mathbf{x}^j_p)}, \tag{2}$$

where Ω^j_t is set of nearest neighbors of $\mathbf{x}^j_t$, $\tau(l(\mathbf{x}^j_p))$ is the color of $\mathbf{x}^j_p$, $\Delta\epsilon$ is the CIELAB color distance [38] and d is the Euclidean distance. For the ith class, its point distinctness is the sum of all points with the same class label in the scatterplot:

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

where $\delta(l(\mathbf{x}_p^j), i)$ is one if the class label $l(\mathbf{x}_p^j)$ is i and else zero. Similar to [41], we define non-separability as the difference value between $\mathbf{x}_t^j$ with data points belonging to the different classes and same class, thus the contrast to the background can be defined as:

$$\rho(\mathbf{x}_t^j) = \frac{1}{|\Omega_t^j|}\sum_{\mathbf{x}_p^j\in\Omega_t^j} \frac{(1-2\delta(l(\mathbf{x}_t^j), l(\mathbf{x}_p^j)))\Delta\epsilon(\tau(l(\mathbf{x}_t^j)), \mathbf{c}_b)}{d(\mathbf{x}_t^j, \mathbf{x}_p^j)}, \tag{4}$$

the contrast to the background of the ith class is defined as follows:

$$\beta_i^j = \frac{f(\theta_i)}{n_i^j}\sum_p^{n_j} \exp(\rho(\mathbf{x}_p^j))\delta(l(\mathbf{x}_p^j), i) \tag{5}$$

where we use a piecewise function to weight the background contrast:

$$f(\theta_i) = \begin{cases} 1 & \text{if } \theta_i > \kappa \\ -1 & \text{else} \end{cases} \tag{6}$$

κ is a user-specified threshold with the default zero. The reason for the two different weighting schemes is that classes with less or no importance might be treated as the background by viewers [43]. To suppress the saliency of such classes, we introduce a negative importance for them. Since $\rho(\mathbf{x}_t^j)$ might be a negative value, we apply an exponential function to transfer it to positive.

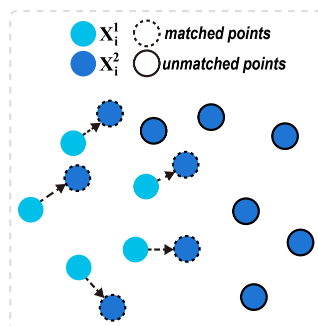


Fig. 2. An one-to-one mapping for computing the changes between two classes.

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

$$H(\mathbf{X}_i^1, \mathbf{X}_i^2) = \min_{\chi}\sum_t d(\mathbf{x}_{i,t}^1, \mathbf{x}_{i,\chi(t)}^2), \tag{7}$$

which constrains an one-to-one mapping χ between points (see an illustration in Fig. 2). This is the classic bipartite matching problem, which can be solved by the Hungarian method [25]. When the number of points of two sets is not equal, we further take the difference between the number of points into account. In doing so, the class change degree is defined as:

$$\theta_i = \frac{H(\mathbf{X}_i^1, \mathbf{X}_i^2)}{\min\{n_i^1, n_i^2\}} + \nu\frac{||n_i^1 - n_i^2||}{\max\{n_i^1, n_i^2\}} \tag{8}$$

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

3.2 Co-Saliency based Color Mapping

On the basis of the co-saliency model, we meet DR1 and DR2 in two ways: co-saliency based color assignment and co-saliency based palette generation.

Co-saliency based Color Assignment. Given a good color palette with P colors ($P \geq M$), the optimal color mapping can be obtained by taking the co-saliency model in Eq. 1 as the objective of the state-of-the-art color assignment method [41]. Starting from a random permutation of P colors, we use the simulated annealing algorithm [1] to find the optimal permutation with two randomized strategies to improve the solution. One is randomly exchanging two colors from the selected m colors and the other is replacing one color from the m selected colors with the one chosen from the unselected $P - M$ colors. With a few iterations, we can obtain a reasonable color mapping as shown in Fig. 1 bottom left.

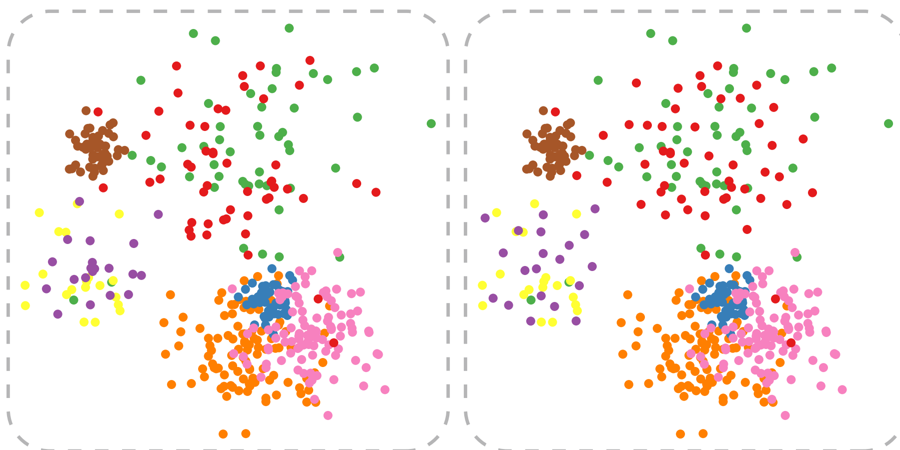


Fig. 3. Visualizing the same data sets as shown in Fig. 4 with the ColorBrewer palette and our assignment method.

However, this method has 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. For example, all colors in the ColorBrewer 8-class Set1 [16] palette are highly discriminable, but it is hard to find a satisfactory solution. Fig. 3 shows an example, where the change of the red class is hard to identify at once even it is very distinctive. Thus, we prompt users to use our co-saliency based palette generation method.

Co-saliency based Palette Generation. 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:

$$\arg\max_{\tau} E(\tau) = \omega_0 E_{CoS} + \omega_1 E_{ND} + \omega_2 E_{CD}. \tag{9}$$

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.

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 ν that is related to the definition of the class change degree which is used as our default class importance. Since ν is fixed in our experiments and the class importance can be specified by user, we mainly discuss the effects of λ and κ.

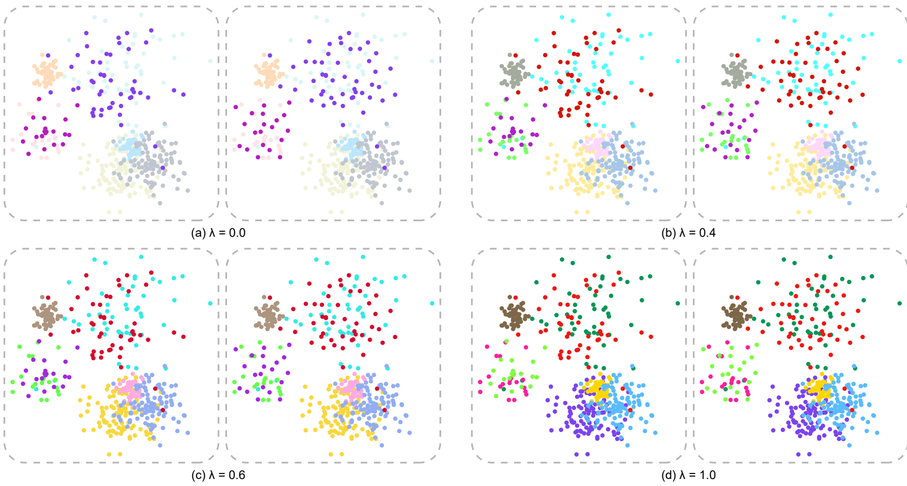


Fig. 4. Effect of λ: (a) result generated by only considering contrast to the background; (b) result generated by setting λ to 0.4; (c) result generated by setting λ to 0.6; (d) result generated by only considering contrast with nearest classes.

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 each 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 (see Sec. 4).

We can observe that when there's only one scatterplot and θ_i of each class is zero, then Equation. 1 is very similar to the objective function of [41]. Our method extends Wang et.al's work to multiple scatterplots with a carefully designed co-saliency model. Besides, we add $\frac{1}{n_i^j}$ to emphasize the class with less points. As shown in Fig. 5(b), with this new term, the little classes, like red, blue and purple classes, become more discriminable.

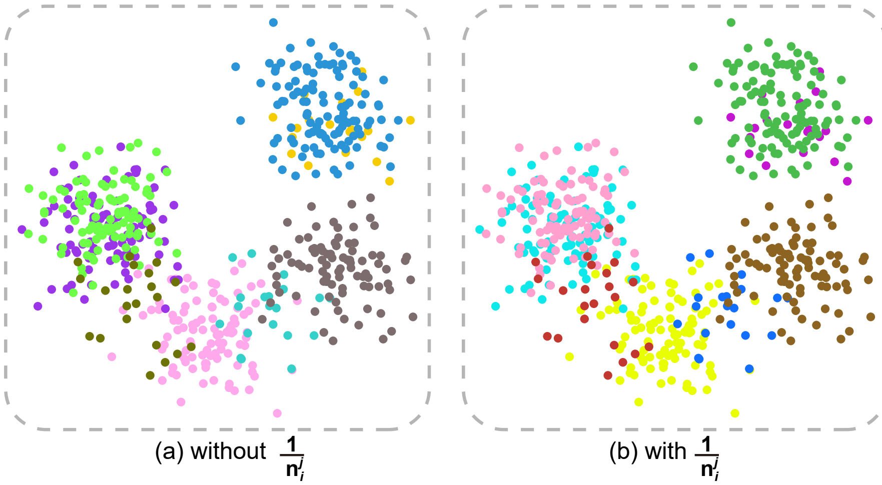


Fig. 5. Effect of $\frac{1}{n_i^j}$: (a) without this term the small classes are hard to catch user's attention; (b) with this term, small classes are easy to find. Palettes are generated with same scatterplot.

Fig. 6. Screenshot of the interactive system. (a) Settings Panel; (b) Control Panel; (c) Visualization Panel; (d) History Panel.

4 INTERACTIVE SYSTEM

To help users interactively design colors for comparing multi-class scatterplots, we developed a web-based multi-view visualization tool [2] (see Fig. 6). It consists of four coordinated views: (a) a settings panel, (b) a control panel for adjusting importance threshold κ and even importance value of each class, (c) the juxtaposed visualizations, and (d) a history view. The control panel shows the decision which classes are highlighted, and the history view allows to quickly explore and access previous color mappings.

[2] https://c3-palette.github.io/

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

Flexible Importance Manipulation. Using θ_i defined in Eq. 1, the classes whose importance values are larger than the threshold κ will be highlighted. Fig. 6(b,c) show an example, where the three classes with the adjusted importance values larger than κ are emphasized with salient *red*, *blue* and *purple* colors, respectively. This control panel allows users to select arbitrary classes of interest to highhlight by simply adjust circle position and κ value. More use cases can be seen in Sec. 6.

Color Vision Deficiency. To help people with a color vision deficiency, we allow users to generate palettes that can be used for different types of vision problem, such as protanomaly and deuteranomaly which result in poor red-green hue discrimination. This is achieved by adopting a color blindness simulator(the source code can be find at github: https://github.com/MaPePeR/jsColorblindSimulator) and then used our matrix for palette evaluation. Fig. 7 show an example, where the left two images show the auto-generated results and the right are the simulated results perceived by people with protanomaly.

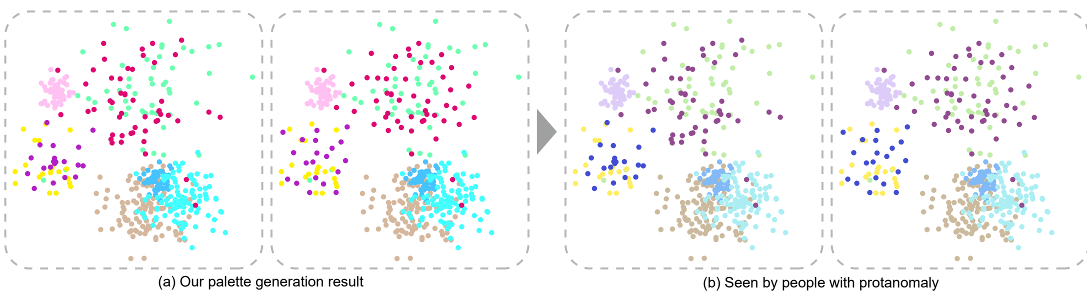


Fig. 7. Exploring the ability of our system to generate palettes for both people with normal vision and color blindness. (a) The automatic generated palette makes the two importance classes with large saliency while maintain good separability between other classes. (b) Simulated results for people with protanomaly. We can see our results maintain a good performance for color vision deficiency.

5 EVALUATION

We evaluated the effectiveness of our method on supporting juxtaposed visual comparisons and the discriminability for reading scatterplots. We conducted two 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 scatterplots:

(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.

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

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

Change magnitude and *Change type*: While the colorization method is the primary independent variable to be investigated, we are also interested in how the effect of different methods would vary based on the level of change between the two scatterplots and the different change type of classes. Thus we first define two types of changes that a class would have across multiple scatterplots: *point number* and *point position*. Then for each change type, we define three levels of change magnitude calculated using Eq. 8: *small*, *medium*, and *large*. (See the next paragraph for the detailed calculation.)

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

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

- *Point number*: For each class in the original scatterplot, we calculated the new point number by multiplying the original number by ($1\pm$ *change ratio*). An addition means to increase the point number, which was implemented by generating the new points with the same distribution as the original class. Subtraction was achieved by randomly deleting data points from the original class.

- *Point position*: Point position contains many types, such as class center position change and shape change. In our experiment, we use the two different position changes mentioned above. For center position change, the center of a class can be moved in a certain *direction* with a specific *distance*. We moved the center towards a random direction by a distance calculated by multiplying a maximal change distance (400 by default) by the *change ratio*. For shape change, we define the shape of a class as the bounding box of its data points. We simulated a shape change of a class by modifying the density parameter of its Gaussian distribution to the opposite direction. For example, a small & dense class ($n = 50, \sigma = 20$) would be changed into a small & sparse ($n = 50, \sigma = 50$) class. In order to produce a new shape for a class, we first calculate the one-to-one mapping between the newly-generated class and the original class using [25] and then linearly interpolated the new point between each two points based on the *change ratio* parameter. We randomly choose one change type when disturbing the class to be changed.

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

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

	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**

5.1 Experiment 1: Spotting the Difference

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

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

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

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

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

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

5.1.1 Experimental Design.

Task & Measures. In this experiment, each participant was asked to perform a *spot-the-difference* task. Inspired by the Spot the Difference game where one needs to compare a pair of similar pictures to detect their differences [11], we asked participants to identify all the classes that have been changed in two scatterplots. At the beginning of each trial, the number of changed classes was provided. Each participant was asked to select all the changed classes by clicking the points belonging to these classes in either of the scatterplots.

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

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

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

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

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

Pilot Study & Power Analysis. We conducted a pilot study involving 28 participants to check the experimental setup and determine the parameters, such as the time limit for a trial. Harnessing by the pilot study, we also obtained our expected effect sizes, which were in further fed into a power analysis. With an effect size Cohen's d of 0.4, alpha level of

Table 2. Participants details for each task.

Task & Group	Spotting the Difference		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

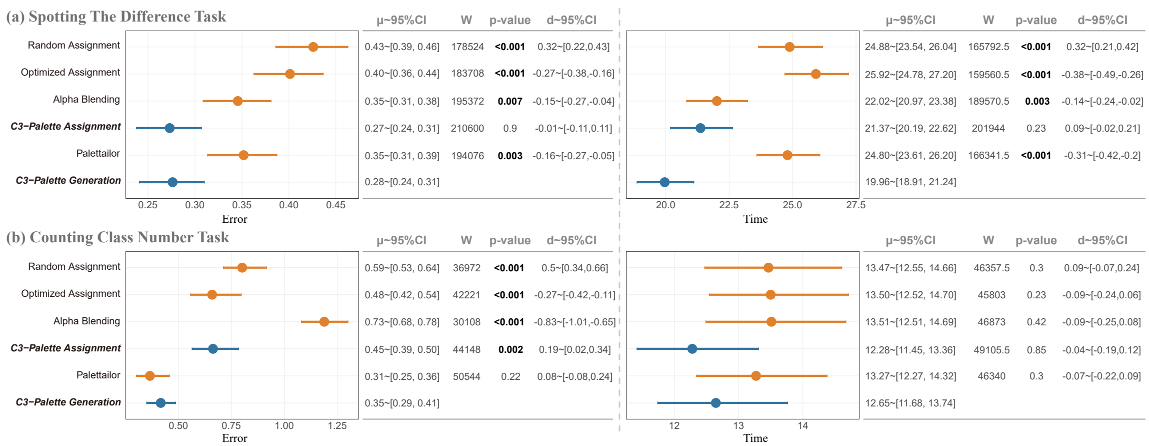


Fig. 8. Confidence interval plots and statistical tables for the two online controlled experiments. Error bars represent 95% confidence intervals. Each table shows the statistical test results of C3-Palette Generation condition with other conditions, including the mean with 95% confidence interval (μ ~95%CI), the W-value and p-value from the Mann-Whitney test, and the effect size (d ~95%CI).

0.05 and beta level of 0.8, the power analysis suggested a minimum number of 100 participants for the spot-the-difference task. See the supplementary material for more details.

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.

5.1.2 *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.8 (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).

Second, we compared error and time with regard to different change magnitudes, and found that smaller magnitude leads to larger error rate and consuming time (as shown in Fig.9 (a) left). This indicates that there exists an significant interaction effect between *change magnitude* and performance, i.e., *change magnitude* would affect user performance. We did the same test to *change type*, the results show that *point number change* is much more difficult than *point position change*(**H3** confirmed).

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

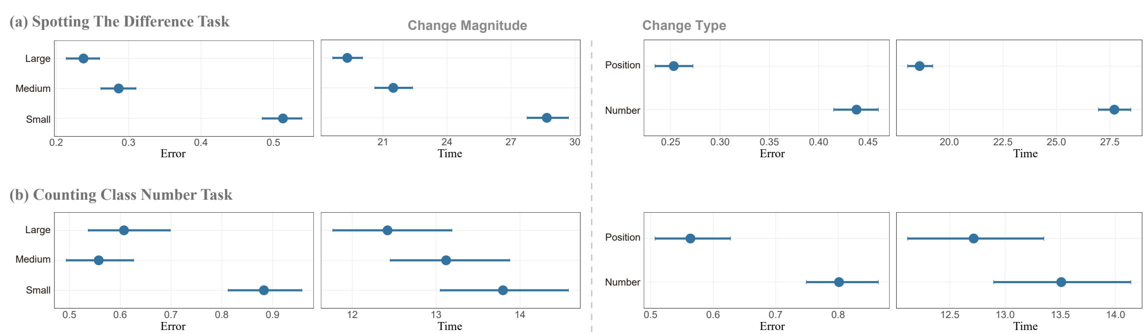


Fig. 9. Confidence interval plots for the two online controlled experiments. (left) Plots for *change magnitude* based on error and time; (right) plots for *change type* based on error and time.

5.2 Experiment 2: Counting Class Number

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

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

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 *Palettailor* 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*).

5.2.1 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.

5.2.2 Results.

Results of this visual separability experiment are shown in Fig.8 (b). Through this study we found that first *C3-Palette Generation* is comparable to *Palettailor* while leads to a significantly lower error rate($p <= 0.001$) 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.9 (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).

5.3 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 comsuming, 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 change magnitude for each change type? Further research is needed to answer these questions so that our approach can be thoroughly evaluated.

6 CASE STUDY

We conducted a case study with a real world data, which is well-known for the use in Gapminder [12], to evaluate the usability of our system.

We choose life expectancy and income as the x axis and y axis, respectively. And we use world regions as the class label. As shown in Fig. 10, due to the limit space, we only show three years. And to make it easy to read, we removed the points with a much larger x value or y value.

We first used the default settings of our system to automatically produce a color assignment result based on Tableau 20 palette for assigning colors to different objects in the dataset, see Fig. 10(a). Since κ is 0 and all the classes are changed, each class is assigned with a salient color to make it more distinguishable. This result is similar to *Optimized Assignment* [41] while our result considers the different importance of classes, i.e., larger importance value has a more salient color. Then we want to explore the two classes with the largest change degree, thus we move the κ control point(the black circle in control panel) to a larger value, as shown in Fig. 10(b). Now we can see the largest changed classes more clearly. But the visual separability between the classes with lower κ value is small, such as the color of

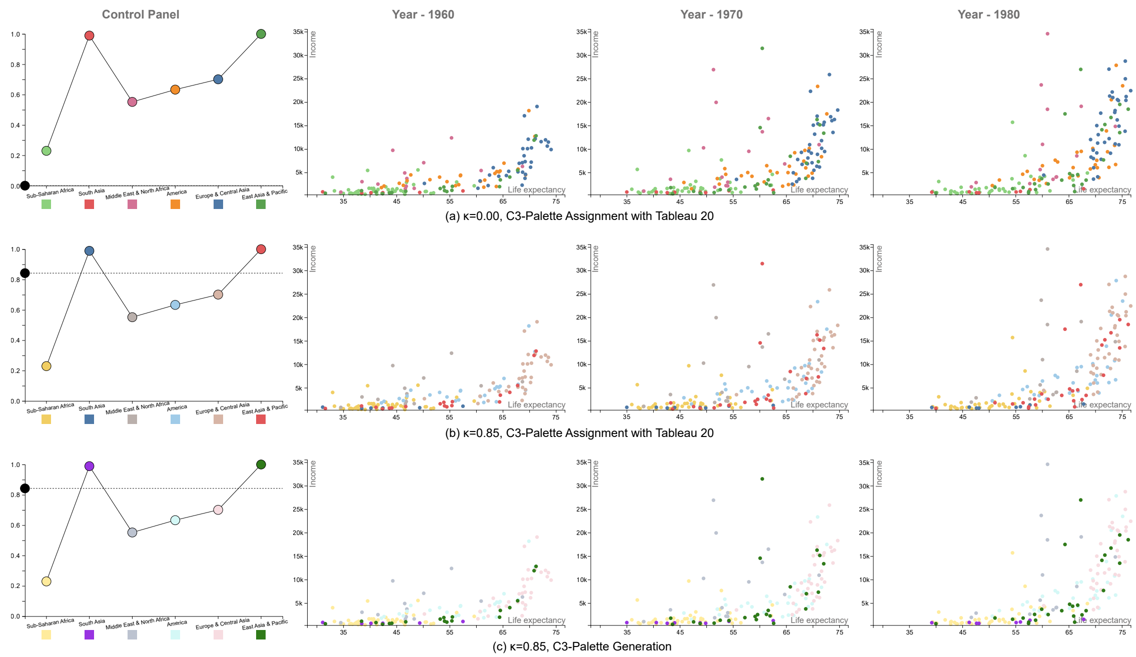


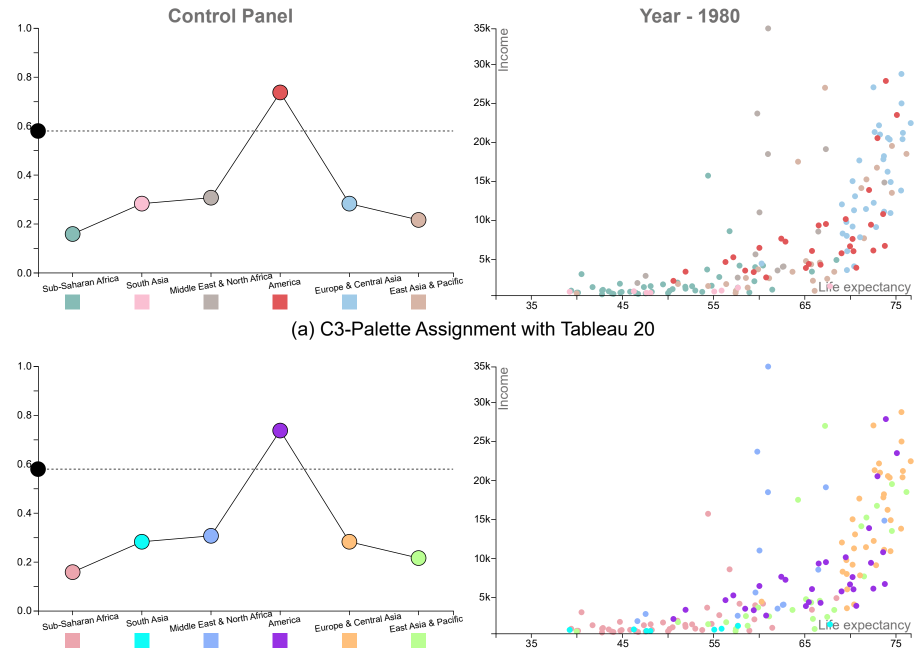
Fig. 10. Gapminder dataset: (a) Result generated by default setting for given palette; (b) User-specified κ value for popping out classes; (c) Automatic palette generation for achieving a better discriminablity.

Middle East & North Africa and *Europe & Central Asia*. We further generate the result by our palette generation method which has a better performance on discriminability, see Fig. 10(c). Through our exploration, we found that *South Asia* should not have a large change degree. This result is caused by our default class importance measure which sets point number change a larger weight in Eq. 8, this is done due to the previous evaluation result that point number change is harder to distinguish than point position change.

Our system also supports manually class importance adjustment, we illustrate this in Fig. 11. For example, we are interested in *America*, thus we can increase the importance value of the corresponding circle and meanwhile, decrease other classes' importance value until lower than κ. We show both assignment result for user provided palette and automatic palette generation result. It's obvious that both results highlight the interested class while palette generation method leads to a much better visual separability between different classes.

7 CONCLUSION

We presented an interactive color design approach for the effective juxtaposed comparison of multiple labeled datasets. It is built upon a novel co-saliency model, which characterizes the most co-salient features between juxtaposed labeled data visualizations while maintaining class discrimination in the individual visualizations. We evaluated this approach in three ways: a numeric study for the class separability in each view, an online study for its usability of detecting changes between multiple views, and a lab study with eye tracking to learn if our approach can alleviate eye movements. The results demonstrate that our produced color mapping schemes are well suited for efficient visual comparison. We further demonstrated the effectiveness of our approach for visually comparing juxtaposed line charts with a case study.



(a) C3-Palette Assignment with Tableau 20

(b) C3-Palette Generation

Fig. 11. Manually define the class importance in the control panel: (a) Result generated based on given palette; (b) Automatic palette generation.

Our work concentrated on juxtaposed comparisons to detect changes between multiple datasets. Although detecting changes is a fundamental visual comparison task, its optimal color palette might not be appropriate for understanding other analytical comparison tasks (such as max delta and correlation tasks [34]. Future work needs to investigate the effectiveness and extensions of our approach for such comparison tasks. Furthermore, our approach produces colors with salient hue to highlight classes with large changes, but those colors do not visually indicate the ranking of class changes. It would be helpful to associate the color ordering constraint [5] with the degree of changes, so that the ranking of class changes can be shown clearly. Last, while we only studied the interaction effect between change magnitude and different colorization methods, we plan to investigate how this effect is influenced by different types of changes, such as point number, center position and shape. The order of rendering is critical for comparison task and we treat it simply in this paper by rendering less important classes first. But when there are multiple important large classes at same positions, the less important class might be overlapped and hard to distinct. Thus a professional render order algorithm is necessary for multi-class scatterplot rendering.

ACKNOWLEDGMENTS

To Robert, for the bagels and explaining CMYK and color spaces.

REFERENCES

[1] EHL Aarts. 1989. A stochastic approach to combinatorial optimization and neural computing. *Simulated Annealing and Boltzmann Machines* (1989).

[2] 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

[3] 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

[4] Ali Borji, Ming-Ming Cheng, Qibin Hou, Huaizu 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] James T Enns. 1990. Three-dimensional features that pop out in visual search. (1990).

[10] 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

[11] 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

[12] Gapminder. [n.d.]. The Gapminder visualization system. https://www.gapminder.org/data/.

[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. Elmqvist, 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. Sedlmair, O. Deussen, D. Lischinski, Z. Cheng, and Y. Wang. 2021. Palettailor: 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 Guaranteed Visibility. *ACM Transactions on Graphics* 22, 3 (2003), 453–462. https://doi.org/10.1145/882262.882291

[34] B. Ondov, N. Jardine, N. Elmqvist, 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] Gaurav Sharma, Wencheng Wu, and Edul N Dalal. 2005. The CIEDE2000 color-difference formula: implementation notes, supplementary test data, and mathematical observations. *Color Research & Application* 30, 1 (2005), 21–30. https://doi.org/10.1002/col.20070

[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

[40] C. Tominski, G. Fuchs, and H. Schumann. 2008. Task-driven color coding. In *Proceedings of 12th International Conference Information Visualisation*. 373–380. https://doi.org/10.1109/IV.2008.24

[41] Yunhai Wang, Xin Chen, Tong Ge, Chen Bao, Michael Sedlmair, Chi-Wing Fu, Oliver Deussen, and Baoquan Chen. 2019. Optimizing color assignment for perception of class separability in multiclass scatterplots. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 820–829. https://doi.org/10.1109/TVCG.2018.2864912

[42] Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. 2000. Guidelines for Using Multiple Views in Information Visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*. 110–119. https://doi.org/10.1145/345513.345271

[43] Dingwen Zhang, Huazhu Fu, Junwei Han, Ali Borji, and Xuelong Li. 2018. A review of co-saliency detection algorithms: fundamentals, applications, and challenges. *ACM Transactions on Intelligent Systems and Technology* 9, 4 (2018), 1–31. https://doi.org/10.1145/3158674

[44] L. Zhou and C. D. Hansen. 2016. A Survey of Colormaps in Visualization. *IEEE Transactions on Visualization and Computer Graphics* 22, 8 (2016), 2051–2069. https://doi.org/10.1109/TVCG.2015.2489649

A RESEARCH METHODS

A.1 Part One

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi malesuada, quam in pulvinar varius, metus nunc fermentum urna, id sollicitudin purus odio sit amet enim. Aliquam ullamcorper eu ipsum vel mollis. Curabitur quis dictum nisl. Phasellus vel semper risus, et lacinia dolor. Integer ultricies commodo sem nec semper.

A.2 Part Two

Etiam commodo feugiat nisl pulvinar pellentesque. Etiam auctor sodales ligula, non varius nibh pulvinar semper. Suspendisse nec lectus non ipsum convallis congue hendrerit vitae sapien. Donec at laoreet eros. Vivamus non purus placerat, scelerisque diam eu, cursus ante. Etiam aliquam tortor auctor efficitur mattis.

B ONLINE RESOURCES

Nam id fermentum dui. Suspendisse sagittis tortor a nulla mollis, in pulvinar ex pretium. Sed interdum orci quis metus euismod, et sagittis enim maximus. Vestibulum gravida massa ut felis suscipit congue. Quisque mattis elit a risus ultrices commodo venenatis eget dui. Etiam sagittis eleifend elementum.

Nam interdum magna at lectus dignissim, ac dignissim lorem rhoncus. Maecenas eu arcu ac neque placerat aliquam. Nunc pulvinar massa et mattis lacinia.