GVVST: Image-Driven Style Extraction from Graph Visualizations for Visual Style Transfer

Sicheng Song, Yipeng Zhang, Yanna Lin, Huamin Qu, Changbo Wang, and Chenhui Li

Abstract—Incorporating automatic style extraction and transfer from existing well-designed graph visualizations can significantly alleviate the designer's workload. There are many types of graph visualizations. In this paper, our work focuses on node-link diagrams. We present a novel approach to streamline the design process of graph visualizations by automatically extracting visual styles from well-designed examples and applying them to other graphs. Our formative study identifies the key styles that designers consider when crafting visualizations, categorizing them into global and local styles. Leveraging deep learning techniques such as saliency detection models and multi-label classification models, we develop end-to-end pipelines for extracting both global and local styles. Global styles focus on aspects such as color scheme and layout, while local styles are concerned with the finer details of node and edge representations. Through a user study and evaluation experiment, we demonstrate the efficacy and time-saving benefits of our method, highlighting its potential to enhance the graph visualization design process.

Index Terms—AI for VIS, Style Transfer, Graph Visualization, Node-Link Diagram, Visual Saliency, Human-computer Interaction.

1 Introduction

Graph visualization plays a pivotal role in data representation, enabling the display of topological relationships between various entities [1, 2], such as social networks [3], biological systems [4], and transportation networks [5]. Specifically, our focus in this paper is on node-link diagrams [6], which are a fundamental technique in information visualization for representing relationships by explicitly drawing nodes and the links between them [7]. To create an engaging and informative graph visualization, designers devote considerable attention to visual styling, ensuring that the end product is both aesthetically appealing and easy to comprehend [8, 9]. The internet hosts numerous well-designed graphical visualizations, often presented as static bitmap images. While some approaches [10, 11] attempt to embed visual style codes within these images, yet existing images cannot be applied. This poses challenges to designers who wish to adapt these exemplary styles into new graph visualizations, as it necessitates the cumbersome task of recreating and refining the original code manually. Thus, the extraction and transfer of visual styles from bitmap images have become a crucial research area, seeking to develop efficient and effective methods for repurposing existing design styles in the creation of new graph visualizations.

- *S. Song, C. Li, and C. Wang are with the School of Computer Science and Technology, East China Normal University, E-mail: scsong@stu.ecnu.edu.cn, {chli, cbwang}@cs.ecnu.edu.cn.*
- *Y. Zhang is with Institute of Education, Tsinghua University. E-mail: yp-zhang22@mails.tsinghua.edu.cn*
- *Y. Lin, and H. Qu are with the Hong Kong University of Science and Technology. E-mail: ylindg@connect.ust.hk, huamin@cse.ust.hk*
- *This work was supported by the NSFC under Grants (No. 62072183 and 62102278), the NSSFC under Grant 22ZD05, and the Shanghai Committee of Science and Technology, China (Grant No. 22511104600).*
- *Corresponding authors: Chenhui Li and Changbo Wang.*

Graph visualization has witnessed substantial advancements in recent years, with numerous techniques [12, 13, 14] being developed for extracting raw data from various types of visualizations. Significant attention has been devoted to chart data extraction, which involves recovering raw data from chart images such as bar charts [15], pie charts [16], and line charts [17]. These basic chart types have seen the emergence of both heuristic [18] and deep learning methods [19] for data extraction. Despite these successes, graph visualization data extraction remains more challenging due to the presence of numerous data attributes and styles. Prior works have laid the groundwork for reverse visualization, focusing on extracting topological relationships [20, 21]. In contrast, our work emphasizes the extraction of graph styles, including color, layout, node and edge representations. In other words, we focus on style transfer for graph visualizations.

Within the computer vision community, style transfer is often studied as an extension of the texture synthesis problem [23, 24], involving the extraction and transference of texture from a source to a target. Nevertheless, visualization images possess unique characteristics that markedly distinguish them from natural images. Therefore, the emphasis of style transfer in the visualization community is particularly on extracting the style template of the chart [25]. Chart style is a critical aspect of visualization, and extracting style from existing visualizations for transfer to other visualizations is an important research direction. Prior works on chart style transfer have primarily focused on color extraction, employing heuristic methods [26] and deep learning models [27, 28] for various visualization types. Moreover, specific style transfer techniques have been developed for particular visualization types, such as pictorial [29] and bar chart [30] visualizations. In this paper, we concentrate on significant styles in graph visualization and propose the development of pipelines for extracting these styles using novel methods, further advancing the field of graph

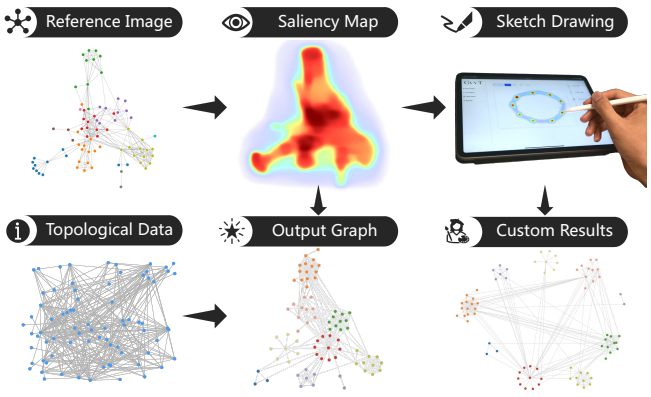


Fig. 1. GVVST is an image-driven style transfer technique that facilitates the process of constructing new visualizations by transferring the reference graph style to a given set of graph structures. GVVST also allows users to modify the graph layout by simply sketching their desired design, thereby providing greater flexibility and customization in the visualization process. The topological data in the figure is from Les Miserables [22].

visualization style transfer.

We first conduct a formative study to investigate the styles that individuals focus on in graph visualization. Through user interviews with 29 volunteers who have over 2 years of experience in visualization research or visual design, we identify several recurring themes that emerged as crucial aspects in the design process, including layout, color schemes, representations of nodes and edges, and labels and legends. Based on the insights provided by the participants, we aggregated the most significant styles as global and local style features in graph visualization. Our research aims to provide a solid foundation for further exploration in the field of graph visualization style transfer.

We present a novel pipeline **GVVST** (Graph Visualization Visual Style Transfer) to transfer the visual style of an existing graph visualization image to another graph while preserving its topological structure. The method consists of two main components: global style transfer and local style transfer. The former focuses on aspects such as color scheme and layout, while the latter is concerned with the finer details of node and edge representations. The training dataset is generated using D3 [31], and the generator employed in prior work is adapted to add control over the graph style. The global style transfer employs a saliency detection model for layout extraction and a K-means-based color extraction method for color transfer. On the other hand, the local style transfer utilizes a multi-label extraction neural network capable of processing visualization images as input and producing a 10-dimensional feature vector as output. The resulting local style is stored as a JSON file for the interactive interface to access and utilize both styles.

We provide a human-centered interactive interface for graph style transfer, allowing users to modify the visual style of a graph image based on automatically extracted styles. This method caters to various display aspect ratios, irregular shapes, and artistic layout generation. Results show that our approach outperforms prior work in accuracy and efficiency for graph style extraction.

Our contributions include three aspects:

(1) We conduct a formative study to understand user concerns and define a new problem for style transfer in graph visualization.
(2) We develop an end-to-end pipeline that effectively extracts global and local styles from graph images.
(3) We build interactive user interfaces based on our method and demonstrate the effectiveness of our method through evaluations and user study.

2 Related Work

Our work builds on previous research along three aspects: Sec. 2.1 Graph Visualization Design, Sec. 2.2 Chart Data Extraction, and Sec. 2.3 Visual Style Transfer.

2.1 Graph Visualization Design

Graphs are important mathematical structures that represent relationships between entities. While alternative representations for graphs like trees [32] and matrix [33] exist, this paper specifically focuses on node-link diagrams [6] as the primary visualization method for graphs. This approach provides flexibility and an intuitive representation of the graph's structure. When creating node-link diagrams, designers take into account various visual properties, such as layout and color, to enhance their aesthetic appeal and facilitate effective communication of the underlying data [8, 9].

The graph drawing and network visualization community has been researching this for years and deriving best practices to support network analysis [34, 35, 36]. Recent advances in machine learning offer novel perspectives on graph layout. Notably, methods such as those proposed by Kwon et al. [37] and GND [38] integrate graph kernels and neural networks to enhance layout estimation. DeepDrawing [39] and subsequent neural models [40, 41] further demonstrate the potential of deep learning in this domain. In graph drawing, emphasis is placed on optimizing visual clarity and analytical utility by adhering to well-established goals such as eliminating node overlaps, minimizing edge crossings, and optimizing edge lengths [42, 43] to facilitate better interpretation and analysis of the graph structure. While these are established goals in the field, our work does not directly apply these principles, focusing instead on node placement based on salience derived from our style transfer technique. This approach prioritizes the visual prominence of nodes within the graph structure and enables the application of force-directed methods within clusters to align with the conventional goals of graph drawing.

2.2 Chart Data Extraction

Recent advancements in visualization have led to the development of various techniques for extracting data from different types of visualizations [12]. Chart data extraction, which aims to recover raw data from visual representations, is an area that has garnered significant research interest [13, 14]. Research has produced reliable methods for extracting data from common chart types such as bar charts [18, 44], pie charts [16, 45], and line charts [17, 46]. Early approaches, such as those by Savva et al. [18], employed heuristic techniques for data extraction from bar and pie charts. These heuristic methods, while useful for

simple chart types with uniform features, often struggle with generalization and threshold adjustment.

The integration of AI within visualization practices [19] has led to significant progress, particularly with the adoption of deep learning techniques in data extraction workflows [15, 47]. These methods have facilitated a more automated and robust approach, especially for complex chart elements that do not have easily identifiable features, such as the varied slices of pie charts or the details of line charts. Luo et al. [48] have addressed the challenges associated with line charts by focusing on the detection of key data points.

Graph visualizations present a greater challenge for data extraction due to their complexity and the variety of data attributes and styles involved [20]. Initial heuristic-based efforts [49] were limited to single-style visualizations and lacked robustness. Deep learning models have been applied to discern the number of nodes and edges within a graph visualization [50], while Song et al. have combined semantic segmentation with heuristic methods to extract topological information from graph images [21], further enhanced by attention mechanisms for diverse visualizations [20]. Other specialized efforts have focused on particular graph types, such as extracting compounds from chemical diagrams [51] and performing relation regression on scientific graphs [52, 53].

These foundational works have predominantly concentrated on the extraction of raw data—such as bar lengths, pie slice proportions, line chart trends, and node relationships—without considering the visual styles. The users could redesign these topological structures interactively. Compared with these works, our work focuses on extracting the graph visual styles (e.g., color, layout, node and edge representations, etc.) from bitmaps rather than its topological structure. Users could input a different topological structure and map the style extracted from the well-designed graph visualization bitmap to it.

2.3 Visual Style Transfer

Visual style transfer is a technique in computer vision that aims to transform the style of an input image to that of another image, while preserving the content of the original image. Chart style is also an important part of the visualization. A vital research direction is the extraction of the style from an existing visualization image to transfer it to another visualization.

Most work on chart style transfer focuses on colors. Early works are mainly heuristic methods. A classic and effective method is to extract the salient colors after clustering the colors of the image in the color space [26], or to extract the colors by geometric methods after constructing the convex hull in the color space [54, 55]. Most of these methods originate from natural images. For visualizations, an important feature is that the chart may have a color legend, and there is some correlation between the colors. Poco et al. [56] use a heuristic method to extract color mapping in chart images by parsing the color legend. Yuan et al. [27] design a convolutional neural network model to learn a limited number of color schemes in chart images. The latest work also uses Variational AutoEncoder (VAE) [57] model to extract color palettes in infographics. Liu et al. [58] combine deep learning models with heuristics to extract harmonious color palettes from natural images and transfer them to visualization images.

Several style transfer techniques are developed for specific types of visualizations. For instance, Yang et al. [29] present a pipeline for pictorial visualization, which involves extracting icons and patterns to facilitate style transfer. Additionally, Huang et al. [30] propose a deep learning-based approach for bar charts that extracts the position style of the XY axis and the legend.

In recent years, advancements in Artificial Intelligence for Generative Content (AIGC) have introduced novel applications in style transfer [59]. These techniques offer the benefit of not being limited to predefined styles. However, they fall short in allowing users to control the quality of the generated output, and their results, being bitmap-based, lack interactivity and adjustability. Our approach diverges from such techniques in two ways: first, our style transfer results are vector-based, allowing for interactive adjustments; second, unlike AIGC methods, which focus solely on transferring image textures without altering the input layout or visualization properties, our approach leans toward transferring style attributes that are pertinent to both the reference image and the visualization, including its layout.

In this paper, we focus on the significant styles in graph visualization and the development of pipelines for extracting these styles based on novel methods.

3 Formative Study

Prior to developing our model, we conducted a formative study to investigate which styles of graph visualization individuals focus on. The styles refers to the visual attributes that are instrumental in crafting a visually appealing and reasonable graph visualization. Our study predominantly involved user interviews with 29 volunteers ($\mu_{age} = 24.9$ years, $STD_{age} = 4.85$, 17 female and 12 male), each having over 2 years of experience in artistic design or visualization research. The participants have at least a foundational level of graph visualization literacy, ensuring they have the necessary skills to comprehend graph visualizations and design graphs. Our research focuses on node-link diagrams, which serve as the primary subject of our investigation. Before initiating the interviews, we dedicated 10 minutes to introduce our study and clarify that the interview content would solely be used for research purposes. We then inquired about the visual styles each participant deemed most significant when creating graph visualizations, allowing them to mention up to five style aspects they prioritized. To facilitate their responses, we provided the participants with 10 exemplary graph visualizations from the D3 [31] Gallery, which they could reference during the interview or base their answers on previous experience. These ten visualizations were selected by two experts engaged in graph visualization research.

During our user interviews, participants provided valuable insights into their preferences and priorities regarding visual styles in graph visualization. Through the analysis of their responses, we identified several recurring themes that emerged as crucial aspects in the design process.

TABLE 1
Interview results for the formative study, with identified styles categorized into global and local. Styles considered in this paper are in bold.

Style Type	Style Aspect	Total	Ratio
Global Style	**Color Scheme**	26	86.67%
	Layout	22	73.33%
	Sparsity	8	26.67%
Local Style	**Node Radius Scale**	13	43.33%
	Edge Width Scale	13	43.33%
	Node Fill	10	33.33%
	Edge Linearity	8	26.67%
	Label Status	7	23.33%
	Legend Location	3	10.00%
	Font Type	2	6.67%
	Node Shape	2	6.67%
	Font Size	1	3.33%
	Arrow Size	1	3.33%

Among these were layout, color schemes, and representations of nodes and edges. Layout encapsulates various user concerns, such as capturing community distribution and strategic node positioning. A good layout allows readers to clearly distinguish nodes, edges, and patterns. Color schemes were deemed essential for visually distinguishing different groups or categories within the graph. Moreover, participants highlighted the significance of carefully selecting node and edge representations to enhance readability and reduce visual clutter. Some participants also mentioned the style of labels and legends, including their fonts and sizes.

We aggregated the number of occurrences of these views as shown in Table 1. We identified two primary categories of user-proposed style features in graph visualization. The first category, referred to as global style, focuses on the overarching characteristics of the graph visualization, such as color theme, layout, and node sparsity. These attributes often shape the initial impression viewers form when encountering a graph visualization. The second category, local style, delves into more granular features, including node fill properties, node size ranges, and edge linearity (straight edge or curved edge). These fine-grained features are highly customizable, as most visualization libraries offer an extensive array of adjustable variables.

Given the potentially infinite variety of styles in graph visualization, it is essential to concentrate our research on the most influential and prevalent features. As such, the style transfer techniques investigated in this paper target the style aspects mentioned by over 20% of the interviewees in our formative study. By emphasizing these features, our research aims to provide a solid foundation for further exploration in the field of graph visualization style transfer.

4 METHODS

Graph style transfer is a process that transfers the visual style of an existing graph visualization image, called the reference graph I_r, to another graph, referred to as the input graph G_i. The objective is to generate a graph visualization I_i that closely resembles the reference graph in terms of style while preserving the topological structure of the input graph.

TABLE 2
The parameters we control in dataset generation. To ensure uniform distribution of style features and prevent model bias, all parameters are randomly assigned within their respective ranges.

Parameters	Ranges
Node Number	[10, 60]
Node Radius	[1, 50]
Node Fill	Solid / Hollow
Community Number	[1, 20]
Edge Number	[30, 75]
Edge Width	[1, 15]
Edge Linearity	Straight / Curve
Label Status	Hide / Show
Label Location	Inside / Outside
Legend Status	Hide / Show
Legend Location	Upper left / Lower left / Upper right / Lower right
Node and Edge Color	R, G, B: [0, 255]
Background Color	R, G, B: [0, 255]
Node Sparsity	[0, 1]
Directionality	Directed / Undirected

Our proposed method consists of two main components: global style transfer and local style transfer. The global style transfer focuses on aspects including color scheme and layout, which have a broader impact on the overall appearance of the graph visualization. The layout refers to the spatial distribution of nodes and edges on the 2D image. On the other hand, local style transfer is concerned with the finer details of node and edge representations, which directly influence the readability and interpretation of individual elements within the graph. This approach ensures that the resulting graph visualization strikes a balance between maintaining the visual appeal of the reference and the logically sound graph representation.

4.1 Training Dataset Generation

Data-driven approaches often necessitate large volumes of labeled training data, which can be challenging to acquire, as manual annotation is both difficult and time-consuming. Recent studies [20, 21, 27, 60] have explored the use of generated datasets as training sets, providing an alternative to manual labeling. Existing graph visualization datasets in prior work [21] primarily focus on extracting topological relationships between nodes, resulting in datasets that are often simplistic and limited in style variation. In contrast, our work aims to extract styles from graph visualizations, necessitating a more diverse and rich set of styles in the training data. To meet this requirement, we adapt the generator employed in prior work [21] by adding control over the graph style.

We utilize D3 [31] for generating the dataset, and during this process, we control the parameters presented in Table 2. These parameters are randomly assigned to generate a uniform distribution that accommodates the style of each case. In terms of graph visualization layout, given the extensive variety of layouts, we allow node positions to be randomly placed on the canvas without overlapping. Concurrently, we introduce a node sparsity parameter that ranges from 0 to 1. This sparsity adds a repulsion factor between the nodes, with higher sparsity values resulting in sparser node distributions. By managing these parameters, our model can effectively learn various possible global and local styles from the dataset.

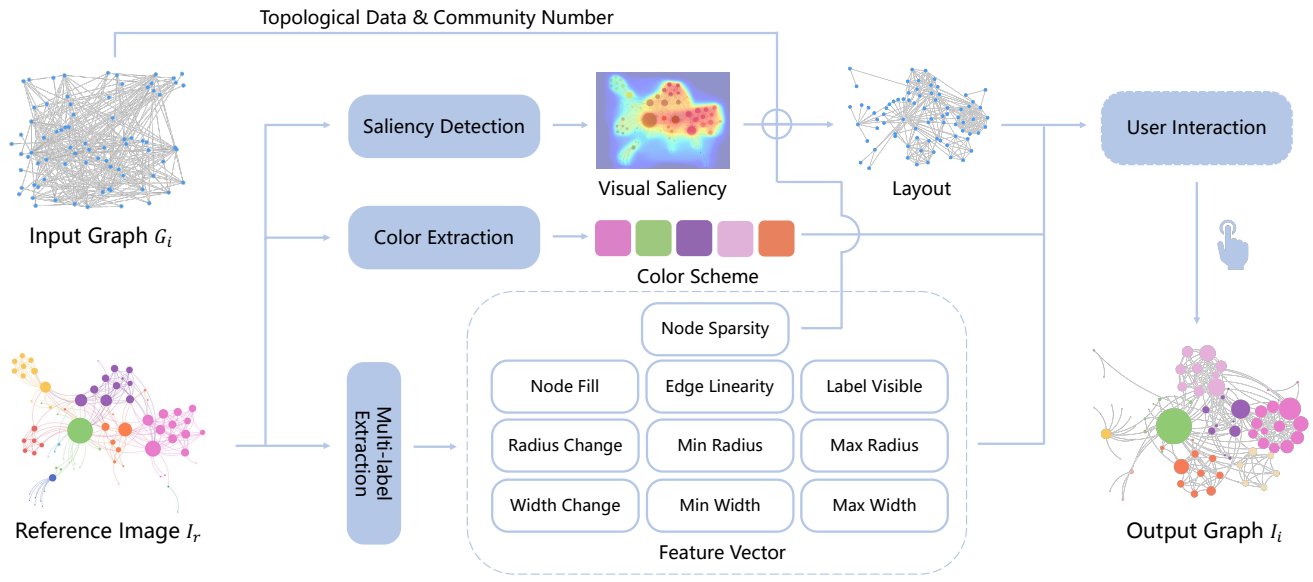


Fig. 2. The pipeline of GVVST including global style transfer and local style transfer.

Ultimately, we generate a training set consisting of $6,000$ images with dimensions of $1,000 \times 1,000$ pixels, accompanied by their respective style labels. Additionally, we create a validation set of $2,000$ images to be utilized during the training process.

4.2 Global Style Transfer

Saliency Detection. Layout plays a pivotal role in graph visualization style, as evidenced by its importance to 73% of participants in our formative study. Through qualitative interviews, we discovered that designers attribute this significance to the layout's profound impact on the global style and its ability to guide readers' attention to critical elements. Drawing inspiration from these findings, we explored the potential of saliency detection models from the field of computer vision as a means to extract layouts effectively.

Saliency detection models have demonstrated success in identifying visually important regions within images, making them a promising candidate for layout extraction in graph visualizations. By adapting these models to the specific context of graph visualization, we can extract meaningful layout information that aligns with designers' intent and expectations. We ultimately fine-tune and employ BASNet [61], a state-of-the-art saliency detection model, for the layout extraction. BASNet is a model of encoder-decoder architecture as shown in Fig. 3. After inputting the graph visualization image of height and width $H \times W$, a saliency map with a resolution of $H \times W$ can be obtained to show the regions that are easy to be noticed by readers in this visualization. In essence, a saliency map is a grayscale image consisting of pixel values ranging from 0 to 255. Pixels with higher values indicate a greater likelihood of capturing the reader's attention, effectively representing the distribution of visual importance across the image.

We use ResNet-34 [62] as the backbone and fine-tune BASNet on the MASSVIS dataset [63]. The input images of any size are resized and normalized to a $224 \times 224 \times 3$ matrix before being fed into the model. Inspired by Zhang et al. [10], we also employ a hybrid loss function $\mathcal{L}_{Sali}$ in our model to acquire more accurate high-level semantic information and preserve low-level details.

$$\mathcal{L}_{Sali} = \mathcal{L}_{BCE} + \mathcal{L}_{SSIM} \tag{1}$$

In this function, $\mathcal{L}_{BCE}$ is BCEWithLogits loss, a combined loss function that merges Binary Cross Entropy (BCE) loss [64] and the logistic sigmoid activation function into a single. $\mathcal{L}_{SSIM}$ represents Structural Similarity Index Measure (SSIM) loss [65], a loss function based on the Structural Similarity Index Measure, which assesses the similarity between two images by considering changes in structural information, luminance, and contrast. The $\mathcal{L}_{BCE}$ is defined as:

$$\begin{aligned}\mathcal{L}_{BCE}(\mathbf{y}, \mathbf{t}) = -\frac{1}{N}\sum_{i=1}^{N}[t_i \cdot \log(sig(y_i)) \\ -(1-t_i) \cdot \log(1 - sig(y_i))]\end{aligned} \tag{2}$$

where $\mathbf{y}$ is the vector of logits output by the neural network, $\mathbf{t}$ is the vector of ground-truth labels that are either 0 or 1, and N denotes the number of samples present in the batch.

The $\mathcal{L}_{SSIM}$ is defined as:

$$\mathcal{L}_{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{3}$$

where x and y are the predicted saliency maps and ground-truth saliency maps, respectively, μ_x and μ_y are the means of x and y, σ_x and σ_y are the standard deviations of x and y, σ_{xy} is the covariance between x and y, and c_1 and c_2 are constants that are added to the denominator to prevent division by zero.

Layout Transfer. In the layout transfer module, we obtain the saliency map of the reference graph visualization through a detection model. Compared to extraction tasks, one of the most challenging aspects of transfer tasks is the requirement for matching. Since the number of nodes and communities between the input graph and the reference graph often differ, we require an automated method to match arbitrary cluster and node quantities while maintaining aesthetics and coherence.

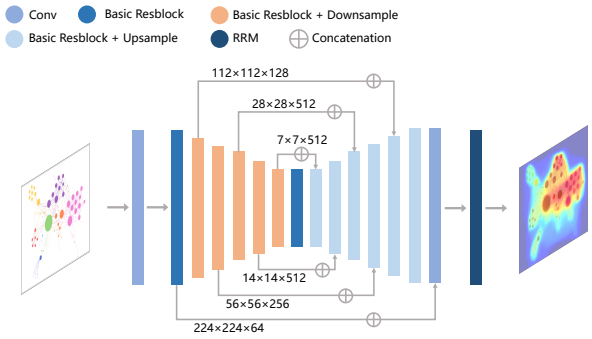


Fig. 3. The visual saliency detection model. This neural network model is an encoder-decoder architecture, and we use ResNet-34 as the backbone to extract features.

Ultimately, we employ a K-means-based clustering approach to cluster the saliency map into k classes, where k corresponds to the number of communities in the input graph. If the input graph does not have a community label, we use the Louvain community discovery algorithm [66] to calculate the community. We use the clustering center of each class as the community center for placing each community of the input graph, matching according to the cluster area size and the product of the node count and node radius within each community. We assign the community with more nodes in the community to the cluster with more salient pixels. After determining the position of each community, we feed all nodes within each community, grouping them community by community, and then generate the node layout for each community as the output. We employ Fruchterman–Reingold layout algorithm [67] within the community for node layout inside the community, where the repulsion factor is obtained from the Node Sparsity feature detected by multi-label extraction model introduced in Sec. 4.3. The saliency-based method makes the community nodes of the input graph distributed in the salient region of the reference image to improve the similarity with the reference graph, while the force-directed algorithm is used in the community to balance the rationality and aesthetics of the layout.

Color Transfer. We also use the K-means-based color extraction method due to its compatibility with the categorical colors commonly found in graph visualizations, as revealed by our interviews with subjects in the formative study. This simple yet effective approach allows for the efficient extraction of categorical color themes, which constitute the majority of color schemes encountered in graph visualizations.

First, we obtain the RGB color values for each pixel within the image, followed by determining the minimum C_{min} and maximum values C_{max} for the R, G, and B. Next, these values are divided into k equal intervals, with k currently set to 3. Each interval is calculated as $len = (C_{max} - C_{min})/k$, resulting in k intervals. Then, the three color channels encompass k^3 intervals.

Each pixel's color is then assigned to the corresponding interval, and the mean value is calculated for every interval. Each mean value is represented as an (R, G, B) tuple, which corresponds to the colors displayed on the color band. The selection of $k = 3$ is based on our observations during the formative study, where we collected well-designed graph visualizations and found no color themes with more than 27 colors. If more colors are needed, a larger value of k can be selected. The extracted color with the highest proportion is deemed the background color, while the other colors are assigned to each community in the input graph, sorted by the number of nodes, according to their respective proportions in the original image.

4.3 Local Style Transfer

Multi-label Extraction Model. In our formative study, we discover that the representation of nodes and edges, commonly referred to as the local style, significantly contributes to the overall aesthetics and effectiveness of graph visualizations. Given the extensive diversity of local styles in graphs, performing end-to-end extraction poses a considerable challenge. To address this issue, we develop a multi-label extraction neural network capable of processing visualization images as input and producing a 10-dimensional feature vector as output. This feature vector comprises 5 dimensions for regression tasks and 5 dimensions for classification tasks.

As shown in Fig. 4, we use ResNet-50 [62] as the backbone. To handle the mixed task consisting of a 5-dimensional classification and a 5-dimensional regression, we introduce a fully connected layer with ten output units. These output units are further connected to ten classifiers. To accommodate both classification and regression tasks simultaneously, we apply a sigmoid activation function to the output of the classifiers. This design choice enables our model to effectively learn the complex relationships between the input image and the diverse set of tasks, yielding accurate and robust results for both classification and regression objectives. Moreover, the benefit of this design endows the model with scalability. In the future, if there are more local styles or additional tasks to be addressed, the model can be easily extended by adding more classifiers.

During the training process of our model, we utilized a composite loss function that integrates multiple loss components to efficiently optimize the model's performance. The $\mathcal{L}_{Local}$ is defined as:

$$\mathcal{L}_{Local} = \sum_{i=1}^{5} \mathcal{L}_{Reg_i} + \sum_{i=1}^{5} \mathcal{L}_{Class_i} \tag{4}$$

where $\mathcal{L}_{Reg_i}$ represents the loss function of the i-th regression task, and $\mathcal{L}_{Class_i}$ represents the loss function of the i-th classification task. $\mathcal{L}_{Reg_i}$ is the L1 loss function, while $\mathcal{L}_{Class_i}$ is the BCE loss function. The L1 loss function is defined as:

$$L_1(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{5}$$

where y_i is the ground truth for the i-th data, $\hat{y}_i$ is the predicted value for the i-th data, and N is the total number of samples in the batch. We train our model on the training set we introduced in Sec. 4.1.

Local Style Mapping. For the Node Fill feature, when the detection result is solid, the extracted color is assigned as the node's fill color. If the detection result is hollow, the color is assigned to the node's border, while the fill color remains transparent. Regarding the Label Status feature, if

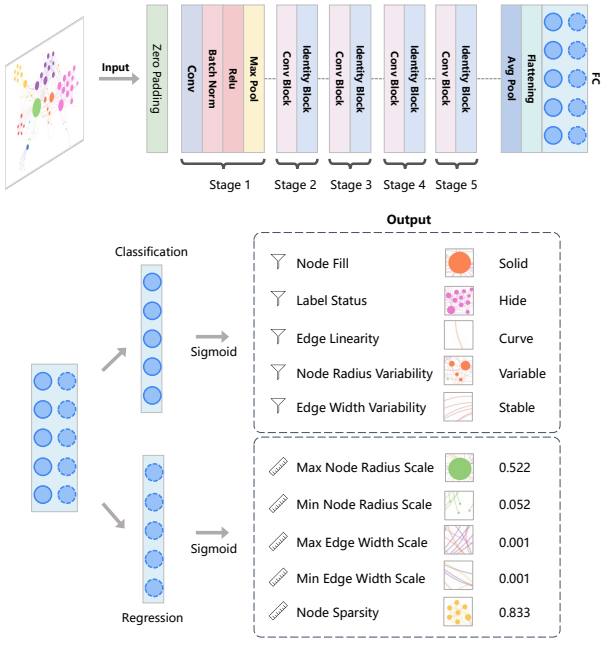


Fig. 4. The feature vector extraction model, which consists of five regression tasks and five classification tasks.

the detection result is affirmative, the label of the input graph is set to be visible. The position for displaying the label is determined based on the average length of the label characters and the average length of the node radius. If the average length of the label characters is less than or equal to the node radius, the label is displayed inside the node; conversely, if the label's average length exceeds the node radius, it is displayed outside the node. In cases where the input graph lacks label information, the node's number is displayed. In the case of the Edge Linearity detection result being classified as a curve, the edge is rendered as a clockwise Bezier curve connecting the start and target nodes.

To determine the node radius, we begin by examining the Node Radius Variable feature. If this feature is detected as variable, we assign the node radius based on the value attribute of the corresponding node in the input graph, mapped to the Node Radius Scale feature interval. In cases where the node in the input graph lacks a value attribute, we utilize the in/out degrees of the node as the value attribute. However, if the Node Radius Variability feature is detected as stable, we set the node radius to the minimum value of the Node Radius Scale.

Regarding the width of the edge, we follow a similar process. First, we assess the Edge Width Variable feature. If it is detected as variable, we determine the edge width by mapping the value attribute of the edge in the input graph to the feature interval of the Edge Width Scale. In instances where the edge in the input graph does not possess a value attribute or if the Edge Width Variability feature is detected as stable, we set the edge width to the minimum value of the Edge Width Scale.

5 Applications

5.1 Automated Graph Design

Recovering Lost Graph Styles. Many graph visualizations on the Internet are disseminated in bitmap format. In academic papers related to graphs, authors often only provide the topology data in the form of JSON or GML files, rather than the full visual representation. Visualization researchers and designers aiming to replicate these meticulously designed graph visualizations must possess adequate coding skills and invest time to recover the underlying style information.

The visualizations in Fig.1 and Fig.2 are derived from visualization-related papers [68, 69], with the underlying data consisting of character relationships in the novel Les Misérables [22] and provided in GML format. These files only contain the graph's topological structure and lack any information about the visualization style attributes. The graph visualizations were designed with the input of designers who considered aspects such as layout, color, and the presentation of nodes and edges. However, when displaying these graphs without the accompanying style data, users who wish to repurpose the visualization for resizing or adjusting the original structure have to recreate the visual style from scratch. Our method excels at automating the task of visual style transfer while upholding aesthetics and similarity, as shown in Sec. 6.3. By extracting global and local styles using our pipeline and combining them with the user-provided topological data, we can transfer the extracted styles into the input topological data, generating a new graph visualization. This process makes recovering the lost styles of graph visualizations simple and efficient.

Coherent Graph Series Design. Another scenario in which users seek automated graph style transfer is when creating a coherent series of graph visualizations. Given that graph visualizations within the same series are often displayed together, ensuring visual consistency with a uniform style becomes pivotal. Users typically find themselves needing to adjust the code of these graph visualizations to ensure consistency, encompassing local style and color in this context.

Our method is adept at accommodating cases where the foundational data of the reference graph diverges from the input topology data. In scenarios involving multi-person collaboration, design collaborators can achieve uniform graph visualization styles by sharing bitmap-format drawings instead of visualization codes. This introduces the potential for extensive large-scale visualization collaborative efforts in the future.

5.2 Sketch-based Graph Design

We have developed an interface that integrates our pipeline. As our style extraction module is based on saliency, users can readily modify the extracted style by employing sketches and user interface (UI) elements, facilitating the rapid creation of customized graph visualizations. This approach offers a seamless and efficient way for users to generate tailored visualizations that cater to their specific needs and preferences, making it a valuable tool in the realm of academic research.

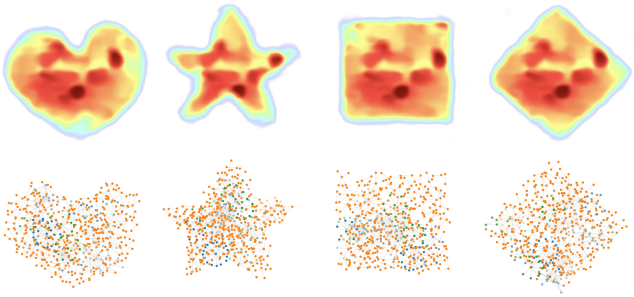


Fig. 5. The first row consists of four saliency maps modified by users, while the second row showcases four results using these modified saliency maps.

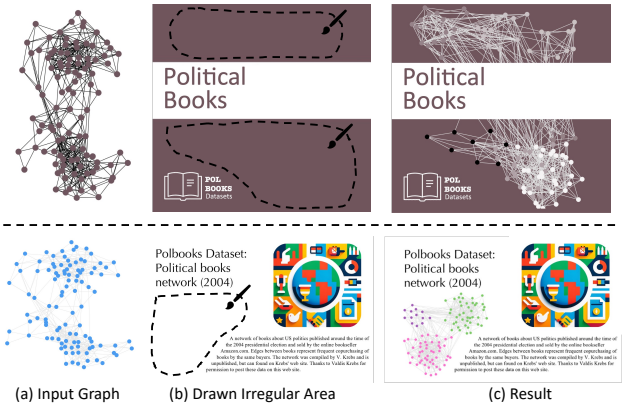


Fig. 6. Our approach adapts the layout of input graphs (a) to irregular regions (b) drawn by users, resulting in graph visualization layouts (c) that fits on the canvas.

Adjusting existing saliency. In our interface, users are provided with a paintbrush and an eraser, which allow them to modify the existing saliency map. As shown in Fig. 5, the user transforms the initial circular salient area into four different shapes: a heart, a pentagram, a square, and a rhombus. Finally, they obtain four graph visualizations with different layouts. That shows our algorithm can change the shape of the layout by modifying the original saliency without changing the original local style and topological data.

Drawing saliency from scratch. Users can also create their own desired saliency entirely from scratch. Our interface provides the option to remove the existing saliency, presenting a clean canvas for customization. On this canvas, users can freely paint the desired saliency levels as needed. The interface accommodates the use of a stylus on a tablet or a mouse on a PC, adjusting the saliency value of the contact area based on the duration of the stroke or click.

As shown in Fig. 1, a user desires to design a circular layout to better observe the relationships between various character communities in a novel character graph. By drawing a circle on the canvas and producing regions of high saliency through multiple clicks or by resting the pen tip in the desired areas, the user can effectively visualize the character community connections. Our system subsequently conducts a saliency analysis based on the user-generated sketch to create an updated layout. The sketch-based graph visualization layout technique introduces innovative applications to the domain of graph visualization design.

Fig. 6 shows two graph visualizations generated by our users in the user study. The users input a graph representing the relationships between political books [70], and then draw an irregular area according to blank areas based on their existing graphic design. This sketch-based approach to creating artistic layouts caters to a wide range of display aspect ratios and irregular-shaped areas, highlighting the versatility and adaptability of our method.

6 User Study

6.1 Recruitment

We recruited 19 participants in our user study. The mean age of the participants was 29.2 years ($STD_{age} = 7.87$, $MIN_{age} = 23$, $MAX_{age} = 51$), with an average experience of 4.5 years ($STD_{exp} = 2.80$, $MIN_{exp} = 2$, $MAX_{exp} = 10$). Our participant pool was diverse, including students in fields like visualization and art design as well as working professionals such as designers and front-end engineers. While they may not all major in graph visualizations, they have had hands-on experience in creating or interacting with them and are familiar with their typical use-cases and representations. This broad range of experience enriches the applicability of our study. Of the 19 participants, 10 were female and 9 were male. They were recruited through social media and all demonstrated a basic literacy in visualization.

To ensure the participants' suitability for the study, we carefully screened their backgrounds and ensured that they possessed the necessary expertise and experience in data visualization. The participants' diverse backgrounds enabled us to obtain a wide range of perspectives and feedback on the effectiveness and usability of GVVST. We informed that the questionnaire results and interview results were only used in our research, and all participants provided informed consent.

6.2 Procedure

In our user study, we bifurcated the evaluation into two parts. In the first part, participants were presented with 8 sets. Each set includes a reference image and a pair of result images that mimic the style of the reference image based on the same topological data. These results were generated using GVVST and a baseline. Given the absence of other automated style transfer tools for graph visualizations, we selected VividGraph [21] as the baseline because its morphological method is a traditional method for data extraction. We leveraged VividGraph to extract data from the original image and then employed our style mapping strategy to achieve the style transfer, addressing an aspect that VividGraph does not cater to. Since VividGraph does not consider graph visualization styling with labels, we introduced OCR technology in the baseline to capture these style elements, ensuring fairness in comparison. For the arrangement of the baseline layout, we employ the automated generation of global force-directed. This approach stands as one of the most frequently utilized algorithms for visualizing graphs. Participants will evaluate the pair of result images using a 5-point Likert scale, focusing on aesthetics and style similarity. The sequence of these images was randomized to ensure unbiased responses.

In the second part, we designed an open-ended graph visualization design task. We provided users with a collection

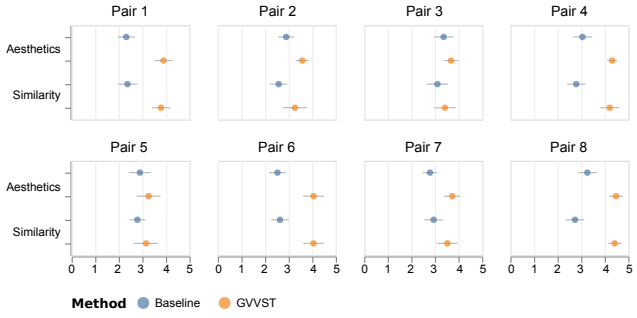


Fig. 7. The results of a 5-point Likert scale on aesthetics and similarity. Circles represent group averages (± 95% confidence intervals).

of 100 well-designed graph visualization images, as well as the topological data for 50 of these images and 16 sets of graph topological data collected by Mark Newman [70]. Users were free to use these high-quality images and employ our system to create similar-styled visualizations for the given topological data. The part was carried out through individual sessions. Before the user study commenced, we briefly introduced the program and obtained participants' consent for recording the study. During the study, we initially familiarized users with GVVST's features and the associated interactive interfaces, ensuring a solid understanding of our research task. Subsequently, participants were asked to utilize GVVST to freely complete the graph visualization creation task, with a 30-minute creation period allotted to each user. Following the hands-on experience, we conducted interviews with participants to discuss their impressions. Drawing inspiration from Notable [71], we devised a 7-point Likert scale questionnaire to appraise the effectiveness and practicality of our system. The questionnaire comprised 10 questions, with Q1 and Q2 addressing the overall functionality and efficiency of our method and system; Q3 and Q4 focused on evaluating the transfer of global styles; Q5 and Q6 targeted the assessment of local style transfer; Q7 and Q8 were dedicated to evaluating our designed interactive interfaces; and finally, Q9 and Q10 aimed to gauge our system's ease of use and retrospectively explore whether our research scope sufficiently covered the discoveries made during formative study.

6.3 Quantitative Results

The results in the first part are shown in Fig. 7. We assign numerical values of 1 to 5 to the options on the five-point scale, with 5 denoting the highest rating. For the baseline method, we obtain the average 2.88 ($\pm 95\%$ CI: $2.63 - 3.04$) on aesthetics, and 2.74 ($\pm 95\%$ CI: $2.51 - 2.89$) on style similarity. While for our methods, we obtain the average 3.88 ($\pm 95\%$ CI: $3.69 - 4.01$) on aesthetics, and 3.73 ($\pm 95\%$ CI: $3.54 - 3.89$) on style similarity. Across all comparisons, our method consistently outperformed the baseline in terms of aesthetics (Related-Samples Wilcoxon Signed Rank Test $Z = 8.39, p < 0.001$) and style similarity ($Z = 8.03, p < 0.001$).

We conduct a more specific analysis of the test results. Regarding aesthetics, the pairs (P1, P4, P6, and P8) exhibit notable improvements, with statistically significant differences between the two results ($Z_1 = -4.61, p < 0.001; Z_4 = -4.24, p < 0.001; Z_6 = -4.90, p < 0.001; Z_8 = -3.58, p < 0.001$). Users found that the layouts, nodes, and edge widths of the results generated by GVVST were more aesthetically pleasing compared to the baseline. For P2, the reference image exhibits a relatively clustered layout with no pronounced clustering, making it challenging for the layout algorithm to demonstrate its advantages. As a result, the difference in aesthetics is smaller in this case($Z_2 = -1.74, p = 0.082$).

We observe results in terms of style similarity for P1, P4, P6, and P8 ($Z_1 = -4.46, p < 0.001; Z_4 = -4.10, p < 0.001; Z_6 = -4.36, p < 0.001; Z_8 = -4.92, p < 0.001$). These questions shared the commonality that GVVST, in comparison to the baseline, extracted local style especially node radius scales have higher accuracy. In the case of P6 and P8, GVVST can extract the style of the curved edges in the reference image, but the baseline method fails. The extracted global style including layouts and node sparsity also displays high similarity to the reference image. For P3, P5, and P7, the difference between our method and the baseline is less prominent ($Z_3 = -1.19, p = 0.233; Z_5 = -1.28, p = 0.201; Z_7 = -1.67, p = 0.095$). We asked users for their reasons. This is because when they assess a graph visualization with labels, their attention is typically directed towards labels. As a result, other disparities between the two graphs are often diminished. This observation is consistent with the results obtained by our saliency detection, which tends to be influenced by and converge to label regions.

The second part results as shown in Fig. 8 provided valuable insights into the users' experience with GVVST. Overall, participants found the system useful (Q1) and efficient in simplifying the graph visualization creation process (Q2). The transfer of global styles (Q3, Q4) and local styles (Q5, Q6) was also perceived as effective, with a preponderance of positive feedback. This suggests that GVVST successfully achieves its goal of producing visually appealing and coherent graph visualizations.

However, it is evident that Q3 and Q4, which focus on the transfer of global styles, received a mixed response. While there were still numerous positive ratings, it must be acknowledged that many participants hold different opinions, particularly for Q4. This suggests that the transfer of global styles may not be as effective or reasonable for some users. The varying responses to Q3 and Q4 could be attributed to individual user preferences, expectations, or the specific graph visualization tasks they undertook. It is possible that some users encountered difficulties in transferring global styles or found the generated visualizations inconsistent with their desired outcomes. We explore this further through interview content in our qualitative results.

Regarding the interactive aspects of GVVST, users found the sketch-based interaction helpful in streamlining their workflow (Q7) and were generally satisfied with the generated styles (Q8). A considerable number of participants strongly agreed that the system was easy to use and learn (Q9), indicating its potential for broad adoption among users with varying levels of expertise. While the system appeared to support a sufficient range of styles for graph visualization (Q10), a small proportion of users remained neutral or somewhat agreed. This feedback highlights potential areas for improvement, such as expanding the available

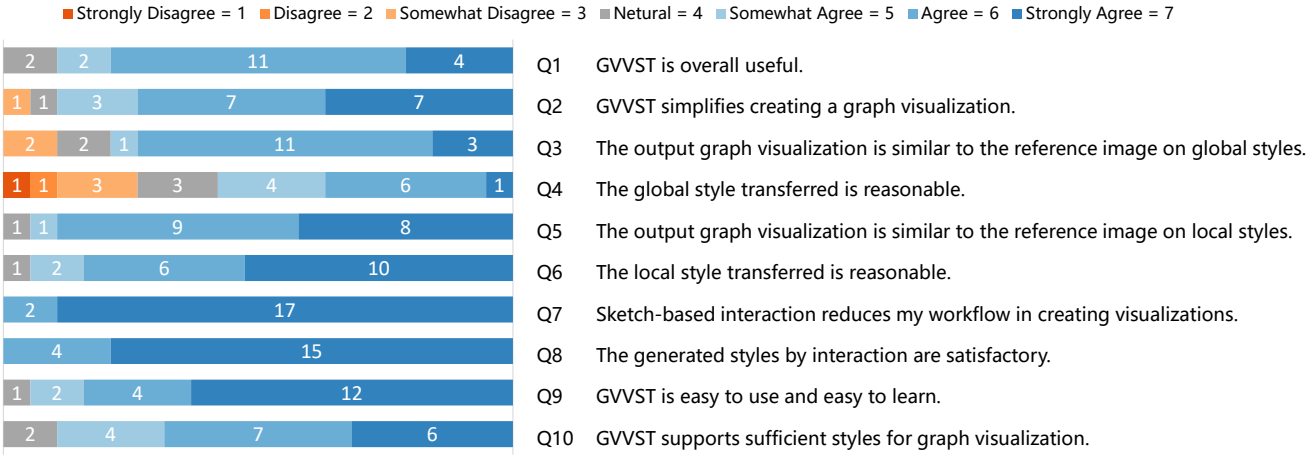


Fig. 8. The results of the second part on our quantitative evaluation of usefulness and effectiveness.

style options to cater to a wider range of user preferences.

Overall, the questionnaire results demonstrate the effectiveness and user satisfaction with GVVST, while also revealing opportunities for further enhancement to better address users' diverse needs.

6.4 Qualitative Results

This section presents the qualitative feedback from participants regarding the image-driven graph visualization style transfer, as gathered during the interview phase. To derive qualitative results, two co-authors independently reviewed the interview records and summarized the participants' feedback. They then discuss and reach a consensus on the common findings revealed during the user study.

GVVST can help users create graph visualizations. In the questionnaire, all participants provided overall positive ratings for GVVST, as they unanimously agreed that it could save time and increase efficiency in many graph visualization creation processes. For instance, two Ph.D students involved in graph mining algorithm research mentioned that they often needed to showcase the topological structures of classic datasets when writing papers, but design style is not their forte. They typically search Google Scholar for related literature using the same dataset and emulate the beautifully designed graph visualization styles for their work. Although they possess relevant programming skills for visualization, they still spend considerable time recreating graph styles based on a bitmap. One of them stated, "GVVST allows me to focus more on my research content while leaving the presentation work to the tool." Furthermore, more than half of the participants expressed their desire for GVVST to be released soon, adding a new graph creating tool to the available resources.

Sketch-based interaction brings new applications to graph visualization design. In our interviews, the sketch-based interaction interface for customizing graph visualization layouts received feedback that exceeded our expectations. Most participants found the interaction both interesting and practical, particularly those involved in design, whose feedback helped us broaden GVVST's range of applications.

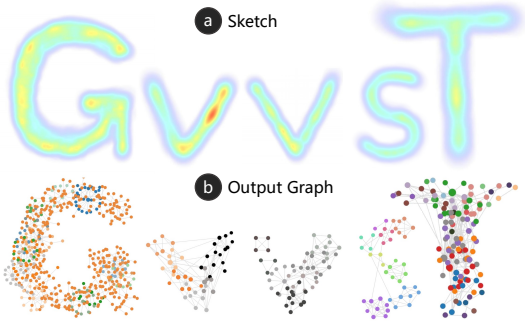


Fig. 9. Users customize the graph visualization layout they need through sketches. The sketch (a) is drawn by the user. Our system outputs a graph visualization (b) whose layout conforms to the sketch.

One graphic designer participant mentioned that although automatic style extraction could save her time, experienced designers often needed to start their designs from scratch. She said, "In fact, adjusting color themes, node and edge representation did not consume much of my time. The critical aspect was modifying the layout to satisfy me, which was a complex process. I love this cute brush stroke interaction." Our interactive interface simplified her workflow by eliminating the need to sketch on paper and then implement the design on a PC. Another participant, a data visualization analyst, said, "I've never felt such control over graph visualization before; the process of hundreds of nodes following my sketch makes me feel like a general commanding soldiers on the battlefield." Additionally, a graph visualization designer used our interaction to draw text shapes and achieved satisfactory results. He noted that the interaction enriched graph visualization layouts with a new visual encoding attribute, presenting words through the layout, akin to word clouds as shown in Fig. 9. Furthermore, this process allowed for flexible control over the drawing process depending on the desired canvas size. Inspired by this designer, we incorporated this application scenario into our applications.

An end-to-end model enhances the user experience. We found that participants' positive attitudes towards GVVST stemmed from the system's prompt data processing. Seven participants mentioned the system's fast processing speed, with younger interviewees showing a higher sensitivity to

time expenditure. One researcher, who specializes in reverse visualization focusing on the extraction of original data from bar charts, noted that he had used some tools for extracting graph data in the past, but they were relatively slow. In contrast, our system impressed him with its responsiveness. We then introduced the model used in our system, and he believed that employing an end-to-end model not only improves algorithm robustness and time efficiency but also significantly enhances user experience. Therefore, we conducted a time performance evaluation experiment in Sec. 7.4 to verify the advantages of adopting an end-to-end model.

Transferred styles will be better with more personalized rules. We noticed that in the questionnaire, Q3 and Q4 received some negative ratings, particularly Q4. We inquired about the reasons for the negative ratings from the participants. One participant who strongly disagreed in Q4 is engaged in data analysis. He stated that he used style transfer to complete a collaboration network graph among scientific researchers. He said, "The topology of the output made it difficult to observe the pattern relationships between communities, even though the style was visually appealing and similar." After completing the automatic style transfer, he interactively adjusted the layout to a circular one, which led to satisfactory results. He also said, "Why can't I choose the style I need? I do not want the layout from references." In fact, it could be easily addressed by adding a UI to control the desired style attributes. In this way, our system can extend beyond making visualizations aesthetically pleasing and contribute to tasks that facilitate the observation of more data features. Another participant who disagreed said, "The current color extraction rules only apply to monochromatic and categorical color schemes. Although most visualizations follow this pattern, the rules fail when encountering graphs not colored based on communities." The current interface allows editing the extracted color bands, but they hoped our system could provide customizable mapping rules. We believe the interviewees' suggestions are constructive and plan to incorporate more personalized interfaces in the future released version.

7 Evaluation

In this section, we conduct a quantitative evaluation of our method. We evaluate global style similarity by analyzing changes in saliency after style transfer and use multiple datasets to demonstrate the robustness of our feature extraction model in extracting styles from graph visualizations with diverse styles across different scales. Finally, we test the time performance of our method to highlight the superiority of the end-to-end approach.

7.1 Test Dataset Composition

Our test dataset comprises two parts. The first part encompasses a real-world dataset named RD, which includes 100 well-designed graph visualization images collected from sources such as the D3 Gallery, E-Charts Gallery, and Google Image. For 54 of these images, we obtain their original data as labels, while the remaining 46 images are manually annotated with their respective style attributes.

The second part comprises additional generated datasets, designed to verify the robustness of our model in handling graph visualizations of varying node and community scales. We select the number of nodes to be less than or equal to 30, 60, 90, and 120. 120 is twice the maximum number (60) of nodes in our training set. The nodes in these four datasets are randomly divided into $[1, 5]$, $[3, 10]$, $[5, 15]$, and $[7, 20]$ communities, respectively. Furthermore, the global and local styles of these graphs are randomly generated. All images in generated datasets have resolutions of $1,000 \times 1,000$. We employ D3 to generate a total of 4 datasets. Each dataset contains 200 images, including 100 images with dark background and 100 images with light background. These datasets, named GD-30, GD-60, GD-90 and GD-120, do not overlap with the training set.

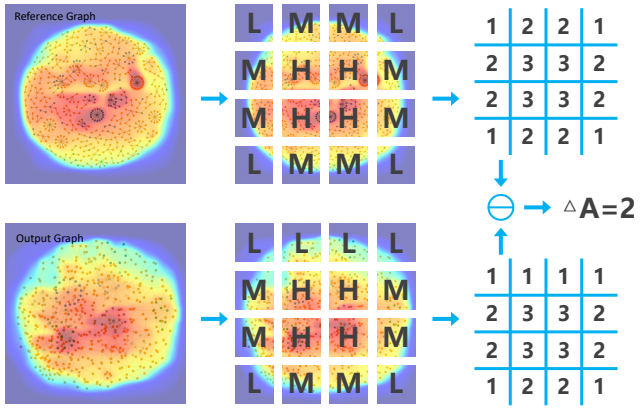


Fig. 10. The global style evaluation model. We divide the reference graph and output graph into a 4×4 grid, calculate the element-wise difference between these matrices, and sum the absolute differences, obtaining a value ΔA.

7.2 Global Style Evaluation

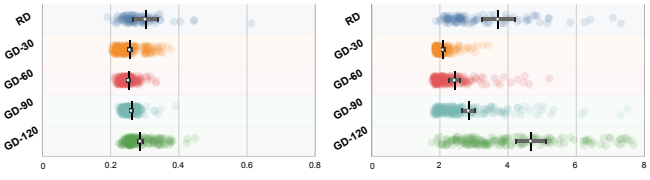
For the global style, we focus on transferring layout and color. To evaluate the similarity of global styles after transfer, we design the following experiment. We input the reference graph's topological data as the input graph G_i into our system for style transfer. Both the reference and output graphs are then input into BASNet to extract their saliency, resulting in the saliency maps S_r and S_o for the reference and output graphs, respectively. We divide these two saliency maps into a 4×4 grid by equally partitioning the height and width, and compute the mean saliency within each cell, where the mean value ranges from 0 to 1. We consider cells with mean values in $(2/3, 1]$ as high-saliency regions, those in $(1/3, 2/3]$ as medium-saliency regions, and those in $[0, 1/3]$ as low-saliency regions. We assign scores of 1, 2, and 3 to low, medium, and high saliency regions, respectively, resulting in two 4×4 matrices A_r and A_o. We compute the element-wise difference between these matrices and sum the absolute differences, obtaining a value $\Delta A \in [0, 32]$. Consequently, we approximate the global style similarity Sim_{gs} between the reference and output graphs as:

$$Sim_{gs} = (1 - \frac{\Delta A}{32}) \times 100\% \tag{6}$$

We quantitatively assess the effectiveness of our approach in preserving and transferring global styles by this evaluation method, including layout and color, from the reference graph to the output graph, which can objectively

TABLE 3
The evaluation of our multi-label extraction model for local style extraction on real-world datasets and generated datasets. We compare our approach with a baseline, VividGraph (a semantic segmentation model combined with morphological analysis). The results demonstrate that our method can extract more styles than VividGraph while achieving higher accuracy and efficiency in tasks they both can complete. Abbreviations: NR Var. = Node Radius Variability, EW Var. = Edge Width Variability, NF = Node Fill, EL = Edge Linearity, LS = Label Status, Min EW = Minimum Edge Width, Max EW = Maximum Edge Width, Min NR = Minimum Node Radius, Max NR = Maximum Node Radius, NS = Node Sparsity.

Methods	Dataset	Classification (Accuracy)					Regression (100 × MSE)				
		NR Var.	EW Var.	NF	EL	LS	Min EW	Max EW	Min NR	Max NR	NS
VividGraph [21]	RD	65%	66%	-	-	-	1.42	9.25	1.78	6.78	-
	GD-30	52%	40%	-	-	-	11.61	15.30	3.01	3.66	-
	GD-60	49.5%	42.5%	-	-	-	9.74	14.41	2.80	2.92	-
	GD-90	43.5%	45.5%	-	-	-	11.62	15.33	2.48	2.81	-
	GD-120	41%	37%	-	-	-	1.64	**0.85**	2.56	2.60	-
GVVST	RD	**86%**	**68%**	**89%**	**97%**	**83%**	**0.97**	**7.85**	**1.80**	**4.90**	**11.33**
	GD-30	**91.5%**	**96.5%**	**100%**	**100%**	**100%**	**0.57**	**0.69**	**0.13**	**0.18**	**9.38**
	GD-60	**100%**	**99.5%**	**100%**	**100%**	**100%**	**1.15**	**1.71**	**0.16**	**0.40**	**3.53**
	GD-90	**98%**	**98%**	**98.5%**	**99.5%**	**98%**	**1.26**	**2.34**	**0.18**	**0.64**	**1.99**
	GD-120	**90%**	**88.5%**	**100%**	**100%**	**99.5%**	**0.62**	3.02	**0.30**	**0.74**	**2.73**



(a) GVVST (b) VividGraph

Fig. 11. Comparison of the time taken by our method and VividGraph under five datasets. The X-axis represents the consumed time (seconds). The colored circles represent each sample. White circles depict group averages with ± 95% confidence intervals.

measure the performance of our style transfer method in maintaining the desired global style.

Since there are only 50 images in the real dataset where we have their original topological data, and it iRs difficult to restore them manually. Therefore, only 50 images with original topological data are evaluated on RD for global style evaluation experiments. Our method achieves an average overall style similarity of 85.13% (T-test $\pm 95\%$ CI: $82.73\% - 97.52\%, p < 0.001$, the same hereinafter) on real datasets. This shows that our layout and color theme are similar to the reference image, which gives people an overall similar feeling on the graphs. At the same time, for the four generated datasets of different scales, we achieved 94.75% ($94.08\% - 95.42\%, p < 0.001$), 93.09% ($92.29\% - 93.90\%, p < 0.001$), 89.20% ($88.24\% - 90.16\%, p < 0.001$) and 84.02% ($82.89\% - 85.14\%, p < 0.001$) Sim_{gs} on GD-30, GD-60, GD-90, and GD-120 respectively. Sim_{gs} has little change and achieved an average value greater than 80%, which shows that our method is robust to graphs of different scales.

7.3 Local Style Evaluation

We quantitatively evaluate the transfer accuracy of local styles in a real dataset and four generated datasets with varying node and community sizes. For the five classification tasks, we employed the accuracy ratio as the evaluation metric, defined as the proportion of instances where the predicted category matches the ground truth. For the other five regression tasks, we select Mean Squared Error (MSE) as our evaluation metric, with lower MSE values indicating better regression performance. The values of all regression tasks have been normalized between $[0, 1]$.

We select VividGraph [21] as our baseline for comparison. VividGraph is a pipeline designed for extracting raw data from graph visualizations, combining an image semantic segmentation model and a morphological analysis algorithm to extract graph topological data. As its morphological analysis also examines node radius and edge width, VividGraph can partially accomplish our local style extraction task. We conduct comparative experiments on tasks that could be executed identically using both methods.

As shown in Table 3, our end-to-end model, GVVST, consistently outperforms the baseline method, VividGraph, across almost all metrics and datasets. In particular, GVVST exhibits significantly higher accuracy in classification tasks, as well as lower mean squared error values in regression tasks. Furthermore, our method is capable of extracting more local styles compared to VividGraph, which highlights the superior effectiveness of GVVST in addressing local style transfer tasks.

7.4 Time Performance

In our user study, we find that users are sensitive to the time the system processes data, so we also evaluate the time performance of our method on various datasets. We evaluate the time consumed to extract styles from a real dataset and four generated datasets, excluding the time dedicated to D3 rendering. We compare with the time taken by VividGraph to extract some styles, while our method transfers all styles.

Utilizing an end-to-end model, our method substantially reduces time cost compared to VividGraph, which integrates morphological analysis. As shown in Fig. 11, our proposed method requires only one-tenth of the time taken by VividGraph and exhibits remarkable stability, with its performance unaffected by variations in image scale. These superior characteristics can be attributed to our adoption of an end-to-end model. This highlights the effectiveness of implementing an end-to-end pipeline and corroborates the decreased computation time reported by users in our user study, ultimately enhancing the experience for many users.

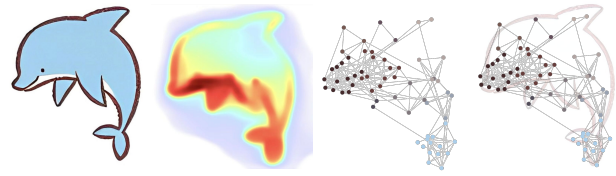


Fig. 12. An artistic background user-generated case. When the user inputs an image of a dolphin as a reference image (a), the color and saliency of the image are extracted (b), and the output is (c). (d) is the overlay of the result and the reference.

8 Discussion

8.1 Aesthetics and Functionality in Image-Driven Graph Visualization

GVVST differs from traditional graph visualization methods in that it attempts to strike a balance between aesthetics and functionality. By transferring the global and local aesthetic elements from existing image-based graphics to the new graphics, GVVST not only emphasizes the visual effect, but also maintains functional integrity, which is achieved by integrating mature algorithmic techniques, such as force-directed algorithms.

Our user study confirms that GVVST's balance of aesthetics and functionality is welcome. This balance can extend applications beyond traditional graph layouts. For example, a user in our research converted dolphin-related graph data [72] into a layout mirroring a dolphin image using our method as shown in Fig. 12, thereby adding a new level of contextual richness to the graph representation. The dolphin-shaped layout generated through GVVST facilitated the identification of graph semantics, which traditional layouts could not easily convey. The visual appeal of the layout does not obscure critical data relationships in this graph. However, in many cases, while such a layout may be aesthetically pleasing, it is not necessarily the most functional layout for data analysis tasks. The research exposed possible conflicts between aesthetics and functionality. This balance is not easy to achieve.

In graph visualization, both aesthetics and functionality are key factors, but how to achieve a perfect balance between the two remains an open question. Our work serves as an initial but significant step towards reconciling the dual demands of aesthetic allure and functional rigor in graph visualization. To address this challenge, future research could explore how to quantify these two factors more effectively or how to automatically adjust the balance between the two through adaptive algorithms. Moreover, the development of metrics for evaluating the impact of aesthetic elements on user engagement and comprehension could refine the design process. This will provide a more comprehensive and fine-grained solution in the field of graph visualization.

8.2 Limitation and Future Work

The current version of GVVST has some limitations in addressing style transfer in graphs. First, when nodes are dense, there is a possibility that the number of colors for the node pixels exceeds that of the background colors. At the same time, the current layout transfer will also cause the overlap of nodes in the restricted space as shown in Fig.5. In light of this, our future work includes refining our approach to better handle such scenarios, aiming to enhance the visual distinction even in dense node situations through advanced post-processing techniques. Second, more style types can be taken into account, such as a wider range of color mappings beyond categorical colors, node distribution patterns, layout symmetry, and classical layout properties. Our future work includes refining our approach to better handle these aspects. Third, we recognize the challenges that accompany automated style transfer, particularly in aligning with user intent and specific graph attributes. While our system currently excels in generating layouts with a broad aesthetic appeal, it occasionally falls short in capturing the specific requirements for certain graph attributes. As a way forward, we envision the incorporation of a human-AI collaborative system. This approach would enable more customized, attribute-based coloring schemes, yet still aim to minimize user effort in the design process.

In the future, we plan to consider more personalized mapping matching rules and support more styles based on user study. We will explore more advanced backbones, such as ViT [73], to replace our local style extraction model and improve its robustness and capacity for classification and regression task dimensions. Additionally, we will apply more interactive techniques [74] to scratch the users' intent for personalization. Furthermore, we plan to release our codes and datasets to support further research in the visualization community.

9 Conclusion

In this research, we introduce GVVST, a novel approach to graph visualization style transfer, leveraging both global and local style extraction techniques and complementing them with an interactive interface. Through a formative study, we identify key aspects of design in graph visualization, which include global and local styles. GVVST then employs a saliency-based model for global style extraction and a multi-label extraction model for local style extraction. We also design the style mapping rules to facilitate the style transfer process, and build user interfaces based on our method. Our user study and quantitative evaluation demonstrate that GVVST can effectively transfer visual styles for graph visualizations in terms of aesthetics and style similarity. The study also reveals limitations, including constrained style types and challenges in function-based layout transfer. However, our work brings a fresh application to the creation of artistic styles for graph visualizations, and signals an initial effort to reconcile the aesthetic and functional dimensions of graph visualization.

References

[1] I. Herman, G. Melançon, and M. S. Marshall, "Graph visualization and navigation in information visualization: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 6, no. 1, pp. 24–43, 2000.

[2] F. Beck, M. Burch, S. Diehl, and D. Weiskopf, "A taxonomy and survey of dynamic graph visualization," in *Computer Graphics Forum*, vol. 36, no. 1. Wiley Online Library, 2017, pp. 133–159.

[3] U. Brandes and D. Wagner, "Analysis and visualization of social networks," in *Graph Drawing Software*. Springer, 2004, pp. 321–340.

[4] N. Gehlenborg, S. I. O'donoghue, N. S. Baliga, A. Goesmann, M. A. Hibbs, H. Kitano, O. Kohlbacher, H. Neuweger, R. Schneider, D. Tenenbaum *et al.*, "Visualization of omics data for systems biology," *Nature Methods*, vol. 7, no. Suppl 3, pp. S56–S68, 2010.

[5] W. Chen, F. Guo, and F.-Y. Wang, "A survey of traffic data visualization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 2970–2984, 2015.

[6] B. Saket, P. Simonetto, S. Kobourov, and K. Börner, "Node, node-link, and node-link-group diagrams: An evaluation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2231–2240, 2014.

[7] Y. Zhao, J. Shi, J. Liu, J. Zhao, F. Zhou, W. Zhang, K. Chen, X. Zhao, C. Zhu, and W. Chen, "Evaluating effects of background stories on graph perception," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 12, pp. 4839–4854, 2021.

[8] C. Ware, H. Purchase, L. Colpoys, and M. McGill, "Cognitive measurements of graph aesthetics," *Information Visualization*, vol. 1, no. 2, pp. 103–110, 2002.

[9] H. C. Purchase, "Metrics for graph drawing aesthetics," *Journal of Visual Languages & Computing*, vol. 13, no. 5, pp. 501–516, 2002.

[10] P. Zhang, C. Li, and C. Wang, "Viscode: Embedding information in visualization images using encoder-decoder network," *IEEE Transactions on Visualization and Computer Graphics*, 2020.

[11] J. Fu, B. Zhu, W. Cui, S. Ge, Y. Wang, H. Zhang, H. Huang, Y. Tang, D. Zhang, and X. Ma, "Chartem: Reviving chart images with data embedding," *IEEE Transactions on Visualization and Computer Graphics*, 2020.

[12] J. Yuan, C. Chen, W. Yang, M. Liu, J. Xia, and S. Liu, "A survey of visual analytics techniques for machine learning," *Computational Visual Media*, vol. 7, no. 1, pp. 3–36, 2021.

[13] K. Davila, S. Setlur, D. Doermann, U. K. Bhargava, and V. Govindaraju, "Chart mining: a survey of methods for automated chart analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[14] F. Bajić and J. Job, "Review of chart image detection and classification," *International Journal on Document Analysis and Recognition (IJDAR)*, pp. 1–22, 2023.

[15] F. Zhou, Y. Zhao, W. Chen, Y. Tan, Y. Xu, Y. Chen, C. Liu, and Y. Zhao, "Reverse-engineering bar charts using neural networks," *Journal of Visualization*, pp. 491–435, 2021.

[16] J. Choi, S. Jung, D. G. Park, J. Choo, and N. Elmqvist, "Visualizing for the non-visual: Enabling the visually impaired to use visualization," in *Computer Graphics Forum*, vol. 38, no. 3. Wiley Online Library, 2019, pp. 249–260.

[17] N. Siegel, Z. Horvitz, R. Levin, S. Divvala, and A. Farhadi, "Figureseer: Parsing result-figures in research papers," in *European Conference on Computer Vision*. Springer, 2016, pp. 664–680.

[18] M. Savva, N. Kong, A. Chhajta, L. Fei-Fei, M. Agrawala, and J. Heer, "Revision: Automated classification, analysis and redesign of chart images," in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, 2011, pp. 393–402.

[19] A. Wu, Y. Wang, X. Shu, D. Moritz, W. Cui, H. Zhang, D. Zhang, and H. Qu, "Ai4vis: Survey on artificial intelligence approaches for data visualization," *IEEE Transactions on Visualization and Computer Graphics*, 2021.

[20] S. Song, C. Li, D. Li, J. Chen, and C. Wang, "Graphdecoder: Recovering diverse network graphs from visualization images via attention-aware learning," *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[21] S. Song, C. Li, Y. Sun, and C. Wang, "Vividgraph: Learning to extract and redesign network graphs from visualization images," *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[22] D. E. Knuth, *The Stanford GraphBase: a platform for combinatorial computing*. ACM Press New York, 1993, vol. 1.

[23] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, "Neural style transfer: A review," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 11, pp. 3365–3385, 2019.

[24] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695.

[25] J. Harper and M. Agrawala, "Converting basic d3 charts into reusable style templates," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 3, pp. 1274–1286, 2017.

[26] Q. Zhang, C. Xiao, H. Sun, and F. Tang, "Palette-based image recoloring using color decomposition optimization," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1952–1964, 2017.

[27] L.-P. Yuan, W. Zeng, S. Fu, Z. Zeng, H. Li, C.-W. Fu, and H. Qu, "Deep colormap extraction from visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 12, pp. 4048–4060, 2021.

[28] L.-P. Yuan, Z. Zhou, J. Zhao, Y. Guo, F. Du, and H. Qu, "Infocolorizer: Interactive recommendation of color palettes for infographics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 12, pp. 4252–4266, 2021.

[29] Y. Shi, P. Liu, S. Chen, M. Sun, and N. Cao, "Supporting expressive and faithful pictorial visualization design with visual style transfer," *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 1, pp. 236–246, 2022.

[30] D. Huang, J. Wang, G. Wang, and C.-Y. Lin, "Visual style extraction from chart images for chart restyling," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 7625–7632.

[31] M. Bostock, V. Ogievetsky, and J. Heer, "D^3 data-driven documents," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2301–2309, 2011.

[32] T. Jankun-Kelly and K.-L. Ma, "Moiregraphs: Radial focus+ context visualization and interaction for graphs with visual nodes," in *IEEE Symposium on Information Visualization 2003 (IEEE Cat. No. 03TH8714)*. IEEE, 2003, pp. 59–66.

[33] M. Ghoniem, J.-D. Fekete, and P. Castagliola, "A comparison of the readability of graphs using node-link and matrix-based representations," in *IEEE Symposium on Information Visualization*. Ieee, 2004, pp. 17–24.

[34] M. Zhu, W. Chen, Y. Hu, Y. Hou, L. Liu, and K. Zhang, "Drgraph: An efficient graph layout algorithm for large-scale graphs by dimensionality reduction," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1666–1676, 2020.

[35] Y. Wang, Y. Wang, Y. Sun, L. Zhu, K. Lu, C.-W. Fu, M. Sedlmair, O. Deussen, and B. Chen, "Revisiting stress majorization as a unified framework for interactive constrained graph visualization," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 1, pp. 489–499, 2017.

[36] J. X. Zheng, S. Pawar, and D. F. Goodman, "Graph drawing by stochastic gradient descent," *IEEE transactions on visualization and computer graphics*, vol. 25, no. 9, pp. 2738–2748, 2018.

[37] O.-H. Kwon, T. Crnovrsanin, and K.-L. Ma, "What would a graph look like in this layout? a machine learning approach to large graph visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 478–488, 2017.

[38] M. Tiezzi, G. Ciravegna, and M. Gori, "Graph neural networks for graph drawing," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[39] Y. Wang, Z. Jin, Q. Wang, W. Cui, T. Ma, and H. Qu, "Deepdrawing: A deep learning approach to graph drawing," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 676–686, 2019.

[40] X. Wang, K. Yen, Y. Hu, and H.-W. Shen, "Deepgd: A deep learning framework for graph drawing using gnn," *IEEE Computer Graphics and Applications*, vol. 41, no. 5, pp. 32–44, 2021.

[41] L. Giovannangeli, F. Lalanne, D. Auber, R. Giot, and R. Bourqui, "Toward efficient deep learning for graph drawing (dl4gd)," *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[42] H. Purchase, "Which aesthetic has the greatest effect on human understanding?" in *International Symposium on Graph Drawing*. Springer, 1997, pp. 248–261.

[43] H. C. Purchase, R. F. Cohen, and M. James, "Validating graph drawing aesthetics," in *Graph Drawing: Symposium on Graph Drawing, GD'95 Passau, Germany, September 20–22, 1995 Proceedings 3*. Springer, 1996, pp. 435–446.

[44] A. Balaji, T. Ramanathan, and V. Sonathi, "Chart-text: A fully automated chart image descriptor," *arXiv preprint arXiv:1812.10636*, 2018.

[45] X. Liu, D. Klabjan, and P. NBless, "Data extraction from charts via single deep neural network," *arXiv preprint arXiv:1906.11906*, 2019.

[46] M. K. I. Molla, K. H. Talukder, and M. A. Hossain, "Line chart recognition and data extraction technique," in *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, 2003, pp. 865–870.

[47] W. Dai, M. Wang, Z. Niu, and J. Zhang, "Chart decoder: Generating textual and numeric information from chart images automatically," *Journal of Visual Languages & Computing*, vol. 48, pp. 101–109, 2018.

[48] J. Luo, Z. Li, J. Wang, and C.-Y. Lin, "Chartocr: Data extraction from charts images via a deep hybrid framework," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 1917–1925.

[49] R. Opmanis, "Optical graph edge recognition." in *VISIGRAPP (3: IVAPP)*, 2018, pp. 184–191.

[50] L. Giovannangeli, R. Bourqui, R. Giot, and D. Auber, "Toward automatic comparison of visualization techniques: Application to graph visualization," *Visual Informatics*, 2020.

[51] M. Oldenhof, A. Arany, Y. Moreau, and J. Simm, "Chemgrapher: optical graph recognition of chemical compounds by deep learning," *Journal of Chemical Information and Modeling*, vol. 60, no. 10, pp. 4506–4517, 2020.

[52] A. Kembhavi, M. Salvato, E. Kolve, M. Seo, H. Hajishirzi, and A. Farhadi, "A diagram is worth a dozen images," in *European Conference on Computer Vision*. Springer, 2016, pp. 235–251.

[53] D. Kim, Y. Yoo, J.-S. Kim, S. Lee, and N. Kwak, "Dynamic graph generation network: Generating relational knowledge from diagrams," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4167–4175.

[54] J. Tan, J.-M. Lien, and Y. Gingold, "Decomposing images into layers via rgb-space geometry," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 1, pp. 1–14, 2016.

[55] J. Tan, J. Echevarria, and Y. Gingold, "Efficient palette-based decomposition and recoloring of images via rgbxy-space geometry," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–10, 2018.

[56] J. Poco, A. Mayhua, and J. Heer, "Extracting and retargeting color mappings from bitmap images of visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 637–646, 2017.

[57] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations*, 2014.

[58] S. Liu, M. Tao, Y. Huang, C. Wang, and C. Li, "Image-driven harmonious color palette generation for diverse information visualization," *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[59] S. Xiao, S. Huang, Y. Lin, Y. Ye, and W. Zeng, "Let the chart spark: Embedding semantic context into chart with text-to-image generative model," *arXiv preprint arXiv:2304.14630*, 2023.

[60] D. Haehn, J. Tompkin, and H. Pfister, "Evaluating 'graphical perception'with cnns," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 641–650, 2018.

[61] X. Qin, Z. Zhang, C. Huang, C. Gao, M. Dehghan, and M. Jagersand, "Basnet: Boundary-aware salient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7479–7489.

[62] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[63] Z. Bylinskii, N. W. Kim, P. O'Donovan, S. Alsheikh, S. Madan, H. Pfister, F. Durand, B. Russell, and A. Hertzmann, "Learning visual importance for graphic designs and data visualizations," in *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, 2017, pp. 57–69.

[64] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.

[65] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, vol. 2. IEEE, 2003, pp. 1398–1402.

[66] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[67] T. M. Fruchterman and E. M. Reingold, "Graph drawing by force-directed placement," *Software: Practice and experience*, vol. 21, no. 11, pp. 1129–1164, 1991.

[68] D. Li, H. Mei, Y. Shen, S. Su, W. Zhang, J. Wang, M. Zu, and W. Chen, "Echarts: a declarative framework for rapid construction of web-based visualization," *Visual Informatics*, vol. 2, no. 2, pp. 136–146, 2018.

[69] S. Song, J. Chen, C. Li, and C. Wang, "GVQA: Learning to answer questions about graphs with visualizations via knowledge base," in *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 2023, pp. 1–16.

[70] M. E. Newman, "Network data," 2013. [Online]. Available: http://www-personal.umich.edu/~mejn/netdata/

[71] H. Li, L. Ying, H. Zhang, Y. Wu, H. Qu, and Y. Wang, "Notable: On-the-fly assistant for data storytelling in computational notebooks," in *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 2023, pp. 1–16.

[72] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, "The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations: can geographic isolation explain this unique trait?" *Behavioral Ecology and Sociobiology*, vol. 54, pp. 396–405, 2003.

[73] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[74] Y. Lin, H. Li, L. Yang, A. Wu, and H. Qu, "Inksight: Leveraging sketch interaction for documenting chart findings in computational notebooks," *arXiv preprint arXiv:2307.07922*, 2023.

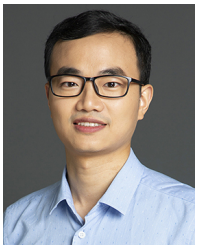
Sicheng Song received his B.Eng. from Hangzhou Dianzi University, China, in 2019. He is working toward the Ph.D. degree with East China Normal University, Shanghai, China. His main research interests include information visualization and visual analysis.

Yipeng Zhang received her B.Eng. from East China Normal University, China, in 2022. She is working toward the Master degree with Tsinghua University. Her main research interests include information visualization and visual analysis.

Yanna Lin is currently a PhD candidate in Computer Science and Engineering at the Hong Kong University of Science and Technology (HKUST). Her main research interests are data visualization and human-computer interaction. She received her BEng from Sun Yat-sen University. For more details, please refer to https://yannahhh.github.io/.

Huamin Qu is a chair professor in the Department of Computer Science and Engineering(CSE) at the Hong Kong University of Science and Technology (HKUST) and also the dean of the Academy of Interdisciplinary Studies (AIS) of HKUST. He obtained a BS in Mathematics from Xi'an Jiaotong University, China, an MS and a PhD in Computer Science from the Stony Brook University. His main research interests are in visualization and human-computer interaction, with focuses on urban informatics, social network analysis, E-learning, text visualization, and explainable artificial intelligence (XAI). For more information, please visit http://huamin.org/.

Changbo Wang is a professor with the School of Computer Science and Technology, East China Normal University. He received his Ph.D. degree at the State Key Lab of CADCG of Zhejiang University in 2006. He was a post-doctor of the State University of New York in 2010. His research interests mainly include computer graphics, information visualization, visual Analytics, etc. He is serving as the Young AE of Frontiers of Computer Science, and PC member for several international conferences.



Chenhui Li received Ph.D. from the Department of Computing at Hong Kong Polytechnic University, in 2018. He is an associate professor with the School of Computer Science and Technology at East China Normal University. He received ICCI*CC Best Paper Award (2015) and SIGGRAPH Asia Sym. Vis. Best Paper Award (2017). He has served as a local chair in VINCI2019. He works on the research of information visualization and computer graphics.