LADV: Deep Learning Assisted Authoring of Dashboard Visualizations from Images and Sketches

Ruixian Ma, Honghui Mei, Huihua Guan, Wei Huang, Fan Zhang, Changye Xin, Wenzhuo Dai, Xiao Wen, Wei Chen

Abstract—Dashboard visualizations are widely used in data-intensive applications such as business intelligence, operation monitoring, and urban planning. However, existing visualization authoring tools are inefficient in the rapid prototyping of dashboards because visualization expertise and user intention need to be integrated. We propose a novel approach to rapid conceptualization that can construct dashboard templates from exemplars to mitigate the burden of designing, implementing, and evaluating dashboard visualizations. The kernel of our approach is a novel deep learning-based model that can identify and locate charts of various categories and extract colors from an input image or sketch. We design and implement a web-based authoring tool for learning, composing, and customizing dashboard visualizations in a cloud computing environment. Examples, user studies, and user feedback from real scenarios in Alibaba Cloud verify the usability and efficiency of the proposed approach.

Index Terms—Dashboard visualization, visual design, stylization, deep learning-based.

1 Introduction

In the last two decades, ubiquitous dashboard visualizations that can monitor all indicators at a glance have triggered phenomenal impact. Recent research pointed out that dashboard visualization provides far more than the sum of its individual parts [1]. However, the generation of dashboard visualizations involves large volumes of input data, rich chart types, and varied usage scenarios. Potential users confront difficulties from the strong dependence on domain knowledge, task-driven visualization design, and holistic implementation.

Most existing dashboard creation tools provide templates for efficient chart initialization and flexible options for expressive customization [2]. However, user choices when utilizing such tools are polarized. On the one hand, visualization novices, who tend to use templates in practice [1], [3], heavily rely on the diversity of templates provided by visualization creation tools. However, a dashboard contains multiple charts, and its design should be determined according to data and analytical tasks, as well as aesthetic considerations, resulting in a vast variety of dashboard designs. Such a variety makes dashboards intractable to cover all situations with a limited number of templates. Novices often fail to present their data with properly designed displays due to their inability to modify the applied templates on demand.

On the other hand, the creation of dashboards from scratch requires iteratively refining the ideas into optimal solutions on the basis of user intention and feedback from actual use. Such designing-and-implementing loop is an upward spiral that gradually migrates from lo-fidelity sketching or prototyping to the complete implementation of the final product. However, existing tools lack support for the quick rendering of dashboard design ideas. The initial conceptualization can only be performed on paper, in which case the evaluation in actual use is skipped, or via a complete implementation, which leads to excessive workloads.

As a result, these difficulties in either expertise or workload still hinder broad audiences from creating and leveraging dashboards despite the growing interest and need for visualization-driven analysis. To bridge the gap between the application of templates and the designing-and-implementing loop, dashboard creators are in need of a tool for dashboard conceptualization — one that can understand user intentions and enable rapid prototyping to try, evaluate, and narrow down possible solutions.

A promising solution is to evaluate ideas through sketches powered by computer-aided chart generation. SketchInsight [4] allows users to sketch on a whiteboard and automatically fills in the details. Recent studies also attempted to provide an experience similar to drawing software, allowing designers to create visualizations with their thinking and working processes [5], [6]. However, such tools are unsuitable for novices because they require users to have a clear understanding of what they want to present. As a useful attempt, tools that recover charts from images try to fill the expertise gap by leveraging exemplars [7], [8]. However, existing methods cannot be applied to dashboards due to the lack of multi-object recognition, consideration of overall layout, and style matching.

In this work, we propose a novel approach called LADV, for efficient prototyping from dashboard images or sketches. This approach allows users to conceptualize their requirements in a lightweight workflow with minimal required expertise and implementation overhead. The kernel of our approach is a novel deep learning-based model that can learn design intention from existing dashboard exemplars, or convert sketches into well-prepared style templates. Our model only learns vital elements from

- *R. Ma, H. Mei, H. Guan, W. Huang, C. Xin, W. Dai, and X. Wen are with the Alibaba Group, Hangzhou China. E-mail: {ruixian.mrx, honghui.mhh, huihua.ghh, mujing.hw}@alibaba-inc.com, {mier.xcy, basi.dwz, ninglang.wx}@taobao.com*
- *W. Chen is with the State Key Lab of CAD & CG, Zhejiang University and is the corresponding author. E-mail: chenwei@cad.zju.edu.cn*
- *F. Zhang is with Senseable City Lab, Massachusetts Institute of Technology.*

Manuscript received July 31, 2019; revised XXX XX, XXXX.

the input instead of precisely extracting all the design specifications from an input image; it then reformulates the extracted information. Reformulation considers the knowledge of the layout and color learned from the collected dashboard instances and manually specified design principles to achieve efficient dashboard designs. The resulting dashboard prototypes can be used to illustrate the design intention and be evaluated for further refinement.

LADV is beneficial for visualization novices and trained designers. With LADV, novices can import dashboard designs by selecting an inspiring example from collection galleries to generate a dashboard template that can be applied and further customized interactively. Moreover, LADV can quickly transform the concepts of trained designers in sketches into ready-to-use dashboard demos, thereby enabling rapid prototyping and evaluation. We also develop an interactive interface with rich, detailed design options to ensure that users can explore and customize the constructed dashboard style and generate dashboards by feeding their data. User studies verify the usability and efficiency of our approach.

In summary, the contributions of this work are twofold:

- This study develops a novel deep learning-based scheme that supports the rapid prototyping of dashboard visualizations from images or sketches;
- The efficiency of our schema is verified with a prototype system that enables efficient creation, customization, and communication of dashboard visualizations.

2 Related Work

2.1 Visualization Creation Tools

Various approaches and tools have been proposed to bring additional accessibility to visualization creation processes from various perspectives, including minimizing implementation overhead, reducing required expertise, and speeding up infographic design [2].

A number of previous studies proposed visual toolkits and grammar of graphics [9] that reduce the complexity of textual programming and improve the efficiency of visualizing data. Such tools help users quickly create visualizations by abstracting over data models and graphical elements ranging from low-level graphic libraries [10], [11], [12] and declarative specifications [13], [14], [15] to high-level chart typologies [16], [17]. Nevertheless, textual programming tools are still nontrivial for visualization novices, who may fall into the tedious trial-and-error process when looking for a reasonable design. By contrast, visualization creation tools with interactive interfaces have been developed to reduce the learning curve for casual users [18], [19], [20]. Most interactive creation tools provide users with visual templates to create charts with drag-and-drop actions and allow customization through direct manipulation on the control panels. Popular commercial software, such as Tableau [21] and Power BI [22] support building dashboards from various chart templates provided. However, the dashboards are not just plain combinations of charts. By contrast, a reasonable dashboard design must consider the organization of charts, overall layout, and consistent styling [23]. In practice, template-based approaches cannot always meet design intentions due to distinctive data modalities, various types of charts, and specific analytical tasks.

An alternative way is to integrate computer-generated data-driven charts within a designer's workflow with drawing tools. Data-Driven Guides [5] allows designers to easily bind data on drawn graphics. Data Illustrator [6] is a vector drawing tool that employs lazy data binding to create expressive visualizations. However, these tools focus on sophisticated infographics designs, which are too time-consuming to be used in the early conceptualization phase. By contrast, sketch-based prototyping enables a high design efficiency. SketchInsight [4] leverages pen and touch to explore and present data on whiteboards, thereby providing efficient initiation for idea evaluation and designing. Similar tools include SketchVis [24] and SketchStory [25]. Although these tools are efficient, they cannot be directly used for authoring dashboard visualizations because they do not consider the overall layout and style consistency.

2.2 Chart Recognition from Images

In creating visualizations, users need to make certain decisions on design styles, such as selecting proper chart types and specifying color mapping. To this end, recent research has turned to "in the wild" visualizations for example-based creation, namely, to extract reusable templates from these visualizations, mostly in the form of bitmaps. A variety of methods for classifying and recovering charts from images have been developed.

Image classification for natural scenes has been widely studied in the machine learning field. A number of methods have been modified and applied to the classification of computer-generated charts [7], [26]. Most chart classification approaches attempt to achieve enhanced recognition accuracy on the basis of artificially defined image features. Savva et al. proposed ReVision [7], which determines the type of chart by using low-level image features, together with the distribution of text regions as an improvement.

After the chart type is identified, additional chart style information, such as textual components, legends, and visual mappings, can be further extracted. The corresponding visual mappings can be identified by localizing the chart's different components, such as axes and legends. Some tools further extract data from images on the basis of the identified information of visual mappings. For example, Revision [7] can extract style and data from bar and pie charts. Poco et al. [27] used an end-to-end method to achieve specialized text localization and extraction on charts. Jorge et al. [8] contributed a method to semi-automatically extract color encoding from visualization images after discrete or continuous color legends are classified and identified. In the study, the legend text was extracted by using optical character recognition methods.

However, this extraction process commonly makes assumptions on the types or structures of charts. Hence, human intelligence can be integrated to improve interpretation accuracy. For instance, ChartSense [28] relies on the manual annotation of textual components, whereas iVoLVER [29] employs gesture-based interactions to identify regions in chart images. These semi-automatic approaches are more adaptable to different charts but are less efficient than others.

Deep learning techniques have been recently used for chart interpretation. Convolutional neural network (CNN) can learn representations of images without specifying the feature extractors [30] and then measure the graphical perceptions [31]. Successful CNN applications in chart interpretation include ChartSense [28] and DeepChart [32]. In this work, we propose a new specialized model for efficient chart interpretation from images containing dashboards.

3 Schema

3.1 Conventional Workflow

We held several meetings with the product managers (PMs), designers, and software engineers of Alibaba Cloud [33] to

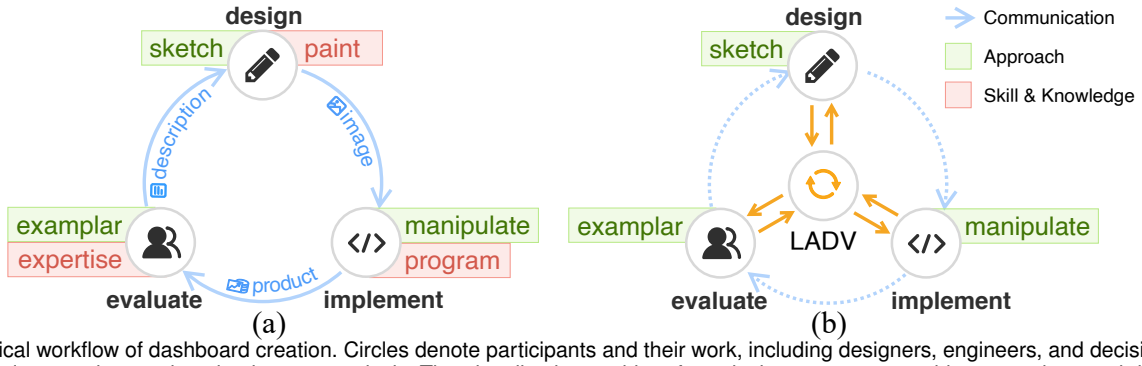


Fig. 1. (a) Typical workflow of dashboard creation. Circles denote participants and their work, including designers, engineers, and decision makers for design, implementation, and evaluation, respectively. The visualization and interface designers are merged because they work in a similar way. Rectangles represent the ways by which different participants finish their work. Red colors (e.g., textual programming) require special skills or knowledge, which incurs implementation overhead and may cause problems for untrained creators. Green colors (e.g., manipulation through interaction) can be easily utilized. Solid lines and arrows indicate the communication between different participants. Everyone must complete their corresponding work before passing the output to the next person in charge to iteratively refine the dashboard design. Such a workflow results in large communication overhead. (b) By contrast, LADV requires minimal skills and knowledge and entails a low workload. Each participant can leverage LADV to complete a lightweight workflow on its own, with minimal required expertise and implementation overhead. This approach can also speedup the prototyping process in the early conceptualization phase.

understand the design and creation of dashboards. Their primary product is a cloud-based dashboard creation tool, DataV (Figure 4), which has generated more than 42,000 dashboards. We discussed elements in designing dashboards, possible problems, and important concerns related to efficiency and effectiveness. Moreover, we also learned the differences between novices and trained dashboard creators. We summarized the conventional workflow of dashboard creation and discussed possible solutions to improve the workflow.

The workflow of **novices** who prefer template-based tools starts from choosing a template to deciding the selection and layout of charts. However, a PM mentioned that *"most users simply apply a template we provide to display their data without making any design modification (on the chart types and layout)."* A large percentage of DataV dashboard instances share a few layouts and colors, which are the provided templates and color palettes. After a template is applied, the creators only make little modifications, such as fine-tuning the positions of charts or trimming the styles (border, texture, etc.). These creators are unable to modify the layouts and color encodings on demand due to their inability to make data-driven or task-based specifications. Accordingly, novices cannot make good use of the rich customization options in DataV to design their own dashboards. The only feasible improvement is to provide novices a large number of templates and color palettes to choose from.

The workflow of **trained creators** can be explained by three steps [34]. First, the data analysts define the project characterization and domain problems and abstract domain-specific tasks to conceive the visual stories from the data. These analysts would probably suggest chart types and graphical elements. Second, the interface designers contribute graphical designs and dashboard layout. Third, the visualization engineers implement all designs. The actual workflow may rollback to form iterative refinement loops (Figure 1(a)). The data analyst and designers tend to conceptualize and make visual design on paper or in the painting software they are used to. These participants often have to wait for engineers to implement the design and then obtain feedback on actual use because the actual generated dashboard often behaves quite differently from the drawn design. For example, a skewed data distribution may lead to unexpected visual effects. However, a complete implementation is seldom necessary. Thus, low-fidelity prototypes for feedback are built in the early stages of the design process, while leaving detailed decisions in future iterations [35], [36]. Inspired by SketchInsight [4] and NapkinVis [37], a tool that enables low- fidelity but fast prototyping from design sketches may help analysts and designers quickly generate and evaluate dashboards without relying on collaboration with engineers. Thus, designers' efficiency can be greatly improved by eliminating implementation and communication overhead.

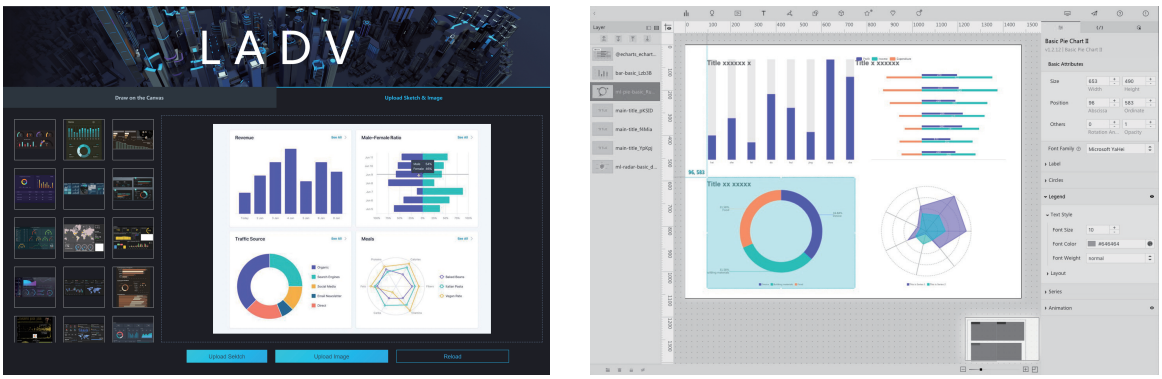
3.2 LADV Workflow

The present study aims to help a broad audience conceptualize dashboard solutions. Our target is not to generate the final product but to come up with a lightweight method for fast prototyping. Prototypes can help users consolidate their ideas and provide a medium for users to express, exchange, and evaluate ideas with others. In designing LADV, we are concerned about the mechanism by which to bring in and inspire users' design intention. However, continued implementation must also be enabled.

We consider two aspects. On the one hand, novices should create dashboards in an expressive way rather than alternating between a few templates. On the other hand, trained creators should reduce implementation overhead to improve efficiency while exploiting their skills.

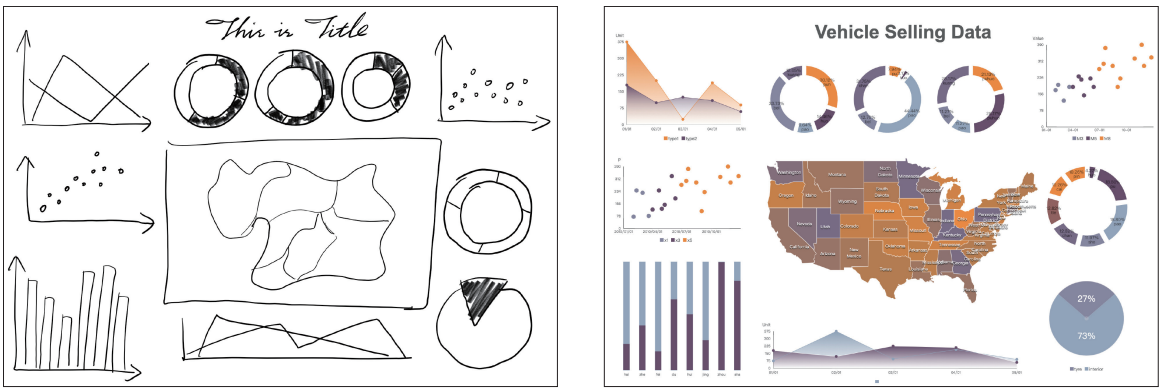
The core idea of LADV is to automatically learn designs from exemplars. Learning from exemplars enables the drawing of inspiration from existing dashboard galleries and modifying designs to conceptualize ideas. This method can also be modified such that it is applicable to sketches. In this way, LADV is a mixed-initiative system that provides designers with the ability to quickly implement their designs. Providing a lightweight workflow that can be performed with high efficiency is advantageous at the early conceptualization stages (Figure 1(b)). We present two scenarios illustrating the mechanism by which LADV works for different audiences.

Novices' mode. One day, Wendy obtained monthly sales data, containing several different economic indicators in three cities and wanted to make a dashboard to report to her manager the next day. She decided to try some new designs because she was unable to find a desirable template in her dashboard creation tool. Normally, she would ask the IT department to help generate the dashboard, but it was already too late to do so. Thus, she began to try LADV for fast dashboard creation. Subsequently, she



(a) (b)

Fig. 2. (a) After the selected exemplar is uploaded, (b) Wendy rapidly created a prototype for further refinement by simple interactions.



(a) (b)

Fig. 3. Creating a dashboard from a sketched image using LADV. (a) The bitmap image on the left is a hand-drawn sketch that presents the design intention. LADV leverages a novel deep learning-based scheme to learn dashboard styles (e.g., charts, chart types, and layout) from the sketch. (b) The dashboard visualization on the right can be automatically constructed by feeding a sample dataset into the learned style.

searched for dashboard images on various networks, such as Google and Pinterest, and found one that fitted her needs (Figure 2(a)). Such image contained a radar graph that could compare the three cities using different indicators. She only needed to upload the image to LADV. Afterward, she was able to generate a dashboard (Figure 2(b)) that had been converted into a template. Thereafter, she imported the template into the dashboard creation tool and applied further customization by interactively linking data and modifying the styles.

Experts' mode. Peter is a dashboard designer and is currently assigned to design a dashboard for a vehicle company. As the vehicle company sells nationwide, he thought that a map could help people effectively understand the national sales data. He also planned to use a line graph to show sales trends and other charts to present different data dimensions. However, he was hesitant on where to place this map and the other charts in the dashboard. Hence, he drew five sketches with different layouts in the painting software. Thereafter, he uploaded the five sketches to LADV to generate dashboards. Finally, Peter selected the one wherein the map chart was in the center of the dashboard (Figure 3).

3.3 Design Considerations

We identified several design considerations by combining our goal with visual design guidelines and dashboard design practices to guide the design of the model and interface. On the basis of these considerations, we built LADV as a tool to recognize and quickly synthesize dashboard templates from exemplars.

C1. Leverage successful designs. The dashboard design is relative complex; thus, taking inspiration from existing work is beneficial [38]. Novices should choose a visible instance as an initialization because doing so is more feasible than providing flexible design approaches. Therefore, users are allowed to initialize their designs by providing images of existing dashboard samples.

C2. Fast prototyping in one step. Users of the dashboard creation tool generally lack the skill, interest, or time to perform complex operations. Hence, we shape LADV such that is as easy as applying a template. Thus, creators can quickly obtain a demo product and then evaluate their design decisions.

C3. Focus on the layout and color palettes rather than on the detailed specifications. Our goal is to allow fast prototyping for the early conceptualize stages. Thus, we do not propose LADV as a tool that can solve all dashboard design details at once. Instead, we focus on the overall dashboard design, including chart type selection, arrangement, and color palettes. Both novices and skilled users regard these concerns as important.

C4. Generate dashboards with aligned layouts and unified color palettes. In automated template generation, the layouts and colors of the resulting dashboards may be defective due to errors or noises in image processing. Novices often produce inefficient designs with such templates. The automatic alignment and unified color palettes can also greatly reduce the cost of subsequent modifications of trained users. We constrain the layout and color with certain rules, which can be manually defined or learned from existing dashboard instances (C1).

C5. Support subsequent customization. Apart from auto-

matic generation, LADV should allow the result to be imported into an interactive tool for further specification and customization of the dashboard. In our prototype system, the template is automatically loaded by the layout and chart editor of DataV, which provides rich customization options for users to modify the dashboard on their demands. This situation allows users to polish their designs, while continuing to generate the final dashboard product.

4 LADV

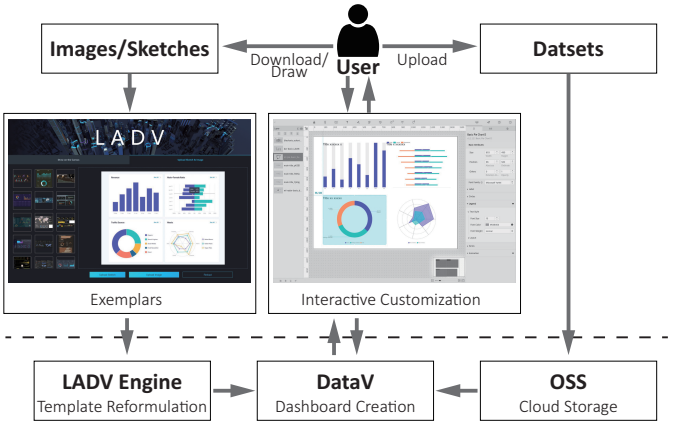


Fig. 4. System architecture of LADV. LADV provides users with a web-based interface for uploading dashboard images or sketches as exemplars, which are reformulated to dashboard templates. In addition, LADV is combined with DataV to generate dashboards and provide rich customization in a cloud computing environment. Moreover, datasets can be stored on the cloud and be easily applied to the generated dashboards.

We implemented LADV as a web application built upon a cloud computing environment. Figure 4 illustrates the major components of LADV and their relationships. The LADV interface consists of a web page for uploading exemplars (Figure 2(a)) and the DataV editor for further customization (Figure 2(b)).

The process of LADV starts by feeding an exemplar into the learning engine (C1). Users can select the exemplar from the given dashboard collection, upload a local image, or draw on a sketch canvas (Figure 5). After the exemplar is uploaded, LADV invokes the backend engine that can learn the charts' designs, layout, and color style (C3). The layout and colors are then optimized on the basis of predefined rules or knowledge extracted from existing instances (C4). Finally, a dashboard template is generated and imported into DataV (C5). With the immediately generated prototype (C2), a user-driven interactive modification can then be performed. Users can also upload their datasets to the Alibaba Cloud Object Storage Service (OSS). The stored datasets can be easily applied to the generated dashboards.

DataV is a mature drag-and-drop dashboard creation production built upon a web-based architecture that allows users to edit and save their dashboards in an online editor without professional knowledge in programming. The DataV editor (Figure 2(b)) consists of four parts: the component panel (top), layer panel (left), canvas (center), and configuration panel (right). The smallest manipulable unit in DataV is the chart, which is called a component. Users can drag and drop to create new components from the component panel containing hundreds of predefined chart templates. Components are listed in the layer panel after creation, and a thumbnail of the overall layout is displayed at the bottom-right corner of the canvas. Users can select, move, and resize components on the canvas by using a mouse. After selection, rich options on the configuration panel are provided to specify the data binding, color encoding, and other styles of the chart.

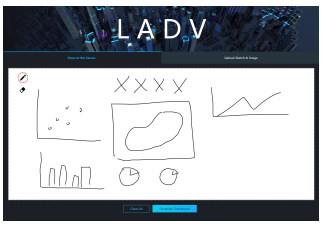


Fig. 5. The sketch canvas provided by LADV for fast convenient sketching.

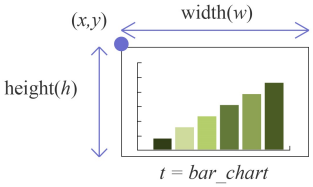


Fig. 6. Annotations on a chart. The position, size, and chart type are annotated.

5 Data Collection

We first collected and annotated an image corpus of 6,348 dashboard instances, including computer-generated ones and hand painted sketches to build LADV. We separately trained two deep learning models for computer-generated images and sketches on the basis of the annotated data. In this section, we first present the mechanism by which we collected and annotated the training data. In the next section, we propose the manner by which to train and use the models.

The computer-generated dashboard images came from the real cases of the DataV platform (Figure 4) and dashboard images crawled from the Internet (2,348 and 2,000, respectively). The sketched dashboard images (2,000) were collected from hand painted drafts by several designers working on dashboard products. We manually selected proper images from the collection for training. All images were normalized to 1920×1080 pixels.

5.1 Annotation

The computer-generated dashboards from the DataV platform were automatically annotated on the basis of the information extracted from their textual specifications. Other dashboards, including images downloaded from the Internet and sketches, were manually annotated. In contrast to the training data for the general image classification model, which only needs a label for each image, each dashboard image contain multiple charts to be individually annotated for an object detection task. The annotation process on DataV dashboards, downloaded images, and sketches are the same. We eliminated charts whose widths or heights were less than 30 pixels. Thereafter, we annotated the chart type, position, and size of each chart. Figure 6 shows that each dashboard consists of several charts, the annotations of which are five-tuples (x, y, w, h, t), where (x, y) is the position of the topleft corner of the chart; w and h are the width and height, respectively; and t is the chart type (e.g., bar or pie chart).

5.2 Computer-Generated Dashboard Images

We classified charts in computer-generated dashboard images into 36 categories, such as basic bar chart, basic line chart, and basic pie chart. The image set was divided into 75% training, 15% validation, and 15% testing. The annotations of all training data for computer-generated dashboard images are denoted as $\mathbf{D}^g$.

5.3 Sketched Dashboard Images

We collected 2,000 sketched dashboard images drawn by the designers or artists by using the painting software or on paper. Sketched dashboards differ from computer-generated ones for the following reasons: the lines are quite irregular, the layout is not well-aligned, and the charts are often not filled with colors. Figure 7 shows some examples. Among the common features of hand-drawn sketches are their roughness and insufficient details. Therefore, we classified charts in the sketched dashboards into 10 categories, each corresponding to one or a group of categories in the computer-generated dataset. For example, the "line chart" in the sketched dataset corresponds to the "basic line chart" and the "line chart with multiple lines" in the computer-generated dataset. Sketched images are divided into 75% training, 15% validation, and 15% testing. The annotations of all training data for the sketched dashboard images are denoted as $\mathbf{D}^s$.

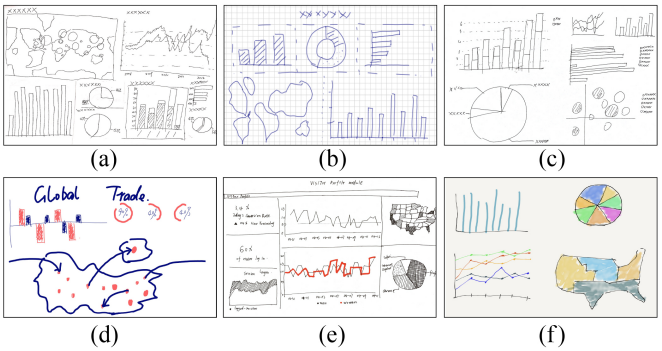


Fig. 7. Examples of sketched dashboard images. The chart styles can be varied. (a–c) are composed of only lines and are not filled, whereas (d–f) are filled with different colors.

6 Approach

In this section, we present the LADV engine, which enables fast prototyping from dashboard images and sketches. We develop a deep learning-based model to recover a dashboard template, which consists of the chart types, layout, and color palette, from the input image. Figure 8 illustrates that LADV takes the following steps to generate a dashboard. After a dashboard exemplar is incorporated into the model, each region that may contain the chart is identified and then verified (Figure 8(a)). The model recommends a color palette on the basis of the colors extracted from the image (Figure 8(b)). The layout of the identified charts is optimized on the basis of a grid-based scheme (Figure 8(c)). Finally, LADV synthesizes a dashboard template from the chart types, color palette, and layout. The generated template is imported into DataV for further customization (Figure 8(d)).

LADV is based on the following approaches:

Chart recognition. We propose a new model based on deep learning technique to recognize charts from dashboard images. This model adapts a Faster R-CNN network [39] that can achieve promising results in object detection. The R-CNN model first checks a certain number of candidate regions that may contain a chart. If a chart exists, then its bounding box and category are predicted. We improve the original Faster R-CNN network to consider the characteristics of different chart types learned from existing dashboards. In this way, LADV can achieve a greatly improved accuracy in recognizing charts from dashboards.

Color palette recommendation. After the charts are identified, LADV begins to specify their colors. The colors are chosen from a color palette extracted from the input image. Instead of individually extracting colors from each chart, LADV creates a global palette for the entire dashboard to ensure visual consistency (C3). LADV uses predefined color palettes for *sketches* because they always lack color information

Layout optimization. The location and size of the charts identified from images are full of random noises, which severely affect the resulting dashboard. We optimize the layout on the basis of a grid design to ensure an effective design (C4).

Finally, a dashboard is generated from the recognized charts, extracted information, and a set of sample data by using DataV specifications. Users can interactively customize the design, import their own data, and evaluate the dashboard in actual use.

6.1 Chart Recognition

6.1.1 Preparation

The input images need preprocessing for optimal results. Borders around the images (often occurring in the case of sketches) can interfere with the recognition of charts and are thus removed by means of a flood-fill algorithm [40]. Wide white margins are also removed to eliminate unusual aspect ratios. Light strokes in sketched images often lead to a decrease in recognition accuracy; thus, we increase the contrast of sketched images.

6.1.2 Candidate Detection

We train two Faster R-CNN models for the computer-generated dashboard dataset $\mathbf{D}^g$ and the sketched dashboard dataset $\mathbf{D}^s$. Moreover, we fine-tune a pretrained network on the basis of VGG16 [28] before the formal training to achieve optimal results The trained Faster R-CNN models can perform object detection tasks on dashboard images. The detection results include a series of chart candidates, their corresponding types, and a *type score* for each candidate that describes the probability of the chart to match the type.

6.1.3 Machine Learning-Based Validation

The conventional Faster R-CNN model cannot achieve desirable accuracy in recognizing dashboard images because of two reasons. First, the color blocks in the background or part of a chart may be misidentified as a separate chart. the characteristics of different chart types are neglected by the model when composing dashboards.

We train a validation model to filter the chart candidates. This task is carried out to deal with such issues and precisely handle various types of charts. The validation model considers not only the *type score* given by the Faster R-CNN model, but also the *learned score* based on the chart characteristics learned from the annotated training data.

Learned characteristics of charts. We store the location information of each chart type in the *computer-generated* dashboard dataset by computing a 3D kernel density estimator (KDE), which considers positions and sizes of the charts. The KDE for chart type T can be calculated in the following steps. Let $\{\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_n\}$ be the descriptors of all charts of type T, defined as $\left\{(x+w/2, y+h/2, \sqrt{w \times h}) \middle| \forall c = (x,y,w,h,t=T) \in \mathbf{D^g}\right\}$, representing the center positions and sizes of charts that are of the same type. The KDE $\hat{f}_T$ is defined as follows:

$$\hat{f}_T(x,y,a) = \frac{1}{nh} \sum_{i=1}^{n} ker\left(\frac{\|(x,y,a) - \mathbf{d}_i\|}{h}\right)$$

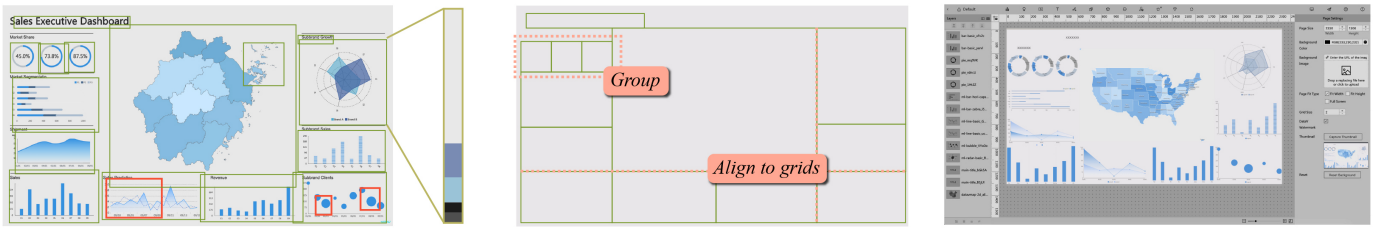


Fig. 8. Pipeline of our approach. (a) Chart candidates are recognized from the dashboard image and then validated through a deep learning-based model. (b) Colors are extracted from the charts based on their statistical information. (c) After optimizing the layout by grid-based scheme, the dashboard is generated and interactively customized (d).

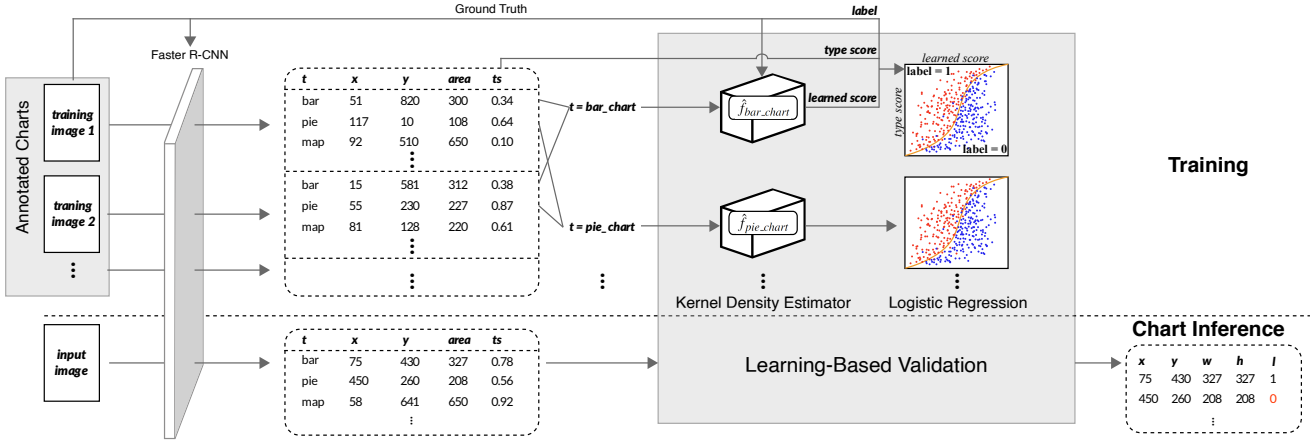


Fig. 9. Deep learning-based validation model. The chart candidates suggested from the Faster R-CNN model are validated to achieve optimal accuracy. The validation model considers the *type score* given by theFaster R-CNN model and a *learned score* based on the chart characteristics learned from the annotated training data. We train the model by fitting a logistic regression of candidates and the ground truth.

where *ker* is the Gaussian kernel, $\|\cdot\|$ is the Euclidean distance, and h is the bandwidth. In our model, h is determined by a grid search algorithm for KDE in [41].

We do not use KDE for the *sketched* dashboard dataset because the chart location distribution in the sketched dashboards is irregular. We assume that people tend to use different layouts when drawing sketches.

Candidate validation model. Next, we train the model for validating candidates by fitting a logistic regression of candidates produced by the Faster R-CNN model and the ground truth from annotated charts.

We denote each chart candidate as $((x,y,w,h,t),(ts,ls),l)$, where (x,y,w,h,t) is the same five-tuple as the chart annotation, ts is the derived *type score* from Faster R-CNN, ls is the *learned score* computed from the learned characteristics and l is labeled using ground truth, computed as follows: For each candidate, we calculate its *learned score* by $ls = \hat{f}_T(x+w/2, y+h/2, \sqrt{w \times h})$, where $\hat{f}_T$ is the $3d$-KDE of type T. We also test if the candidate can match an annotated chart, denoted as $l = 1$ if there is an annotated chart that has the same chart type, and the coincident area of the candidate and the annotated chart is higher than 50% of the annotated chart area; otherwise, $l = 0$.

We train a logistic regression model for each chart type to capture the location and size characteristics (Figure 9). We collect all chart candidates in the training data, filter those whose $ts < 0.1$, and group them by chart type. Thus, a set of regions $\{r \mid r = (D,(x,y,w,h,t = T),(ts \geq 0.1, ls),l)\}$ is formed for each chart type T, and then fitted within a logistic regression model, denoted as

$$h_T(r) = P(l = 1 \mid ts, ls) = \frac{\exp(\beta_0 + \beta_1 ts + \beta_2 ls)}{1 + \exp(\beta_0 + \beta_1 ts + \beta_2 ls)}$$

where β_0, β_1, and β_2 are the parameters determined using maximum likelihood estimation.

6.1.4 Chart Inference

We can choose credible candidates and deal with conflicts between chart candidates on the basis of the trained models. When a dashboard image becomes available, we obtain all its chart candidates from the Faster R-CNN model and filter the regions whose type scores type score are less than 0.1. Given the five-tuple (x,y,w,h,t) and type score ts of each candidate, we calculate $ls = \hat{f}_t(x+w/2, y+h/2, \sqrt{w \times h})$ and $l = h_t(ts, ls)$. We select the reliable chart regions through a greedy algorithm (Algorithm 1) on the basis of the calculated ls for all regions. Nevertheless, a map can serve as the background that overlaps with other charts. The mean average precision (mAP) [42] for recognizing computer-generated images on our test dataset increases from 70.55% to 77.92%. subsection 7.1 provides additional details of the model performance. Our approach is complemented by the prior probability distribution of chart characteristics, which can be integrated with any other machine learning-based model to improve accuracy.

6.2 Color Palette Recommendation

After the charts are identified, the user can choose a color palette to apply. LADV can extract color information from dashboard images

Algorithm 1 Select charts from the candidates.

Input: set of candidate regions R
Output: set of selected regions R_{output}
$R_{output} = \varnothing$
while $|R| > 0$ **do**
  $r_0 =$ the candidate with the largest l value in R
  for all $r \in R, r \neq r_0$ **do**
    if r and r_0 have overlapping parts **then**
      $k = area_{overlap} / \min(area_r, area_{r_0})$
      if $k > 0.3$ **then**
        $R = R \setminus \{r\}$
      end if
    end if
  end for
  $R_{output} = R_{output} \cup \{r_0\}$
  $R = R - \{r_0\}$
end while

to generate a color palette similar to that in the original dashboard. The color extraction is based on the number of pixels occupied by different colors. For sketches that are not colored, we offer predefined palettes that the designers have specified for the user to choose from.

Given that charts in dashboards often contain mixed transparent colors (Figure 10(b)), borders (Figure 10(f)), specified background colors (Figure 10(e)), or other decorations, we do not extract all colors inside each chart. Instead, we focus on the overall style of the whole dashboard for consistency. We extract the "main color" of each dashboard and then generate a color palette around it. Our approach comprises three steps: (1) extracting dominant colors, (2) deciding on the background and text colors, and (3) generating the palette colors.

6.2.1 Extracting Dominant Colors

We first extract the dominant colors. A legend-based color extraction method is not used because the legends in dashboard images typically have low resolutions [8]. We extract the colors of the entire display associated with their proportions. Moreover, we extract 12 dominant colors for each input image by using the median cut algorithm. Each identified color C_i is represented as a four-tuple: three CIELAB channels, and its proportion, namely, $C_i = (L, a, b, p)$.

6.2.2 Deciding on the Background and Text Colors

The background and text colors are indispensable components of a dashboard. LADV selects the color with the largest proportion as the background color. The text color is generated from the identified background color by using a greedy search in the CIELAB color space. Furthermore, the text color is defined as a color with a high contrast to the background color. We use the definition of the contrast ratio of Web Content Accessibility Guidelines (WCAG 2.0)[1]. The contrast ratio is defined as $(L_1 + 0.05)/(L_2 + 0.05)$, where L_1 is the relative luminance of the lighter color and L_2 is the relative luminance of the darker color. LADV identifies a color in the entire color space whose contrast ratio to the background color is larger than 7:1 by using a greedy algorithm [43]. WCAG indicates that such a contrast ratio guarantees an enhanced readability.

1. https://www.w3.org/TR/2008/REC-WCAG20-20081211/#contrast-ratiodef

6.2.3 Generating Palette Colors

LADV generates a color palette with identified dominant colors. LADV removes colors from the dominant colors that are similar to the background or text color. The similarity is measured by the Euclidean distances within the CIELAB color space and the threshold is set to 50. The dominant colors are then clustered with the DBSCAN algorithm [44]. In each cluster, the color that has a large proportion in the cluster is selected. The color palette is generated using clustered colors and then set as the theme color style for the subsequently generated charts.

6.3 Layout Optimization

Figure 8(a) illustrates that the raw output of the recovering model often exhibits a deflective layout. Therefore, we need to adjust the locations of the charts (Figure 8(c)) to ensure that they conform to the original arrangement or to correct the irregular layout of the input image, especially for sketched dashboards that are often inaccurately drawn.

In adjusting the layout, we follow the design regulations, such as the visual balance [45], to achieve aesthetic pleasure. We formulate our design principles on the basis of the grid design guidelines, which are regarded as a systematic approach based on the relationships of alignment [46]. Four steps are sequentially performed.

- *Tessellate using rectangles.* Each chart is a rectangle. The overlapped charts are shrunk until they are no longer overlapping.
- *Align to grids.* Edges that are close in the horizontal or vertical direction are aligned on the same line. However, the moving distance is no more than 50 pixels.
- *Group similar charts.* The charts close to one another and have the same chart type and similar sizes are grouped together. Charts in the same group equally divide the space occupied by the entire group.
- *Set the same margins.* All margins between charts are set to be the same. In practice, we set the margin width to 5 pixels.

6.4 Implementation

We train the Faster R-CNN network by using Pytorch deep learning library [47] and the logistic regression using scikit-learn [41]. Scripts for color extraction and layout optimization are written in Python. The dashboards are generated through the DataV API, which is written in JavaScript.

7 Experiments

7.1 Model Performance

Table 1 shows the mAP, a metric that measures precision and recall, of our models. The Intersection over Union (IoU) threshold is set to 0.5, the learning rate is set to 0.001, and the batch size is set to 4.

Instead of adding the machine learning-based validation for all charts, we compare the validation and Faster R-CNN results of each chart. Thereafter, we only apply the validation for 13 charts, whose scores have been increased. The validation can improve the recognition mAP for computer-generated images from 70.55% to 77.92% across 36 categories by adding the logistic regression models. Table 2) illustrates some improvements. However, the original Faster R-CNN performs well enough on the sketch cases (85% mAP). This outcome is attributed to the relatively limited number of categories (10 different chart types) used in this model.

TABLE 1
mAP values of different models.

Task	Faster R-CNN	With validation
Computer-generated images	70.55%	77.92%
Sketches	85.08%	–

TABLE 2
Top five chart categories with the highest AP improvement.

Chart category	AP improved	Faster R-CNN	With validation
Pseudo 3d world map	50%	50%	100%
Textured bar chart	37.83%	54.55%	92.38%
Line chart	29.29%	67.14%	96.43%
Bar chart	29.10%	67.50%	96.60%
Stacked bar chart	25.79%	66.87%	92.66%

7.2 User Study: LADV Results

We conducted a user study to evaluate the quality of the generated dashboards. This task was performed to investigate the usability and efficiency of LADV. We used only computer-generated images as input to comprehensively evaluate the model results from different perspectives, including chart types, layout, and color.

7.2.1 Study Design

We collected dashboard images grouped by four sources to test the mechanism by which LADV performs on different dashboard styles: (1) Google image search for the term "dashboard", (2) Google image search for the term "Power BI", (3) Google image search for the term "Tableau", and (4) Tableau Public's Gallery. We invited four data analysts to choose two of their preferred dashboard designs in each of the four groups. Accordingly, we collected 32 different dashboard designs. The images were loaded into LADV to generate dashboard prototypes. Figure 10 presents some examples. In each column, the upper image is the original one, and the lower image shows the generated dashboard.

We recruited 16 participants (8 females and 8 males, aged 23–35 years old, mean=26.75, STD=3.1). All participants were confirmed to have no form of known color blindness. The participants were asked to complete a questionnaire, in which each generated dashboard should be rated from different aspects, including aesthetics, willingness to continue to build upon the generated dashboard, types, layout, and color, on a five-point Likert scale ranging from -2 (strongly unsatisfied) to 2 (strongly satisfied). The participants were also encouraged to write down their comments, including the perceived strengths and weaknesses, on the generated dashboards.

7.2.2 Results

Figure 11(a–e) shows the results of the questionnaires (plotted by group). Overall, the participants agreed that the generated dashboard was aesthetically acceptable, except for the group of Tableau Public's Gallery. Some of the generated dashboards received a low aesthetic score. However, the willingness to continue to build upon the generated dashboards remained positive (*"Even if there are some faults ... it can still save time creating a dashboard"*).

LADV received good scores in the type of charts, layout, and color for processing images from the other three groups. However, the score of types for Power BI were relatively low because it contained some chart types, including treemaps and some composite charts, that LADV could not recognize (and DataV had not implemented). The treemaps were recognized as heatmaps (Figure 13(d)) or bar charts (Figure 10(d)). In some other cases, LADV also produced undesirable results. For example, Figure 10(f) (also the outlier of the color score in Figure 11(a)) uses a design that emphasizes the comparison between two groups of charts by using different colors. Our color extraction strategy cannot capture such features, and the resulting dashboard is encoded with the wrong colors.

7.2.3 Discussion

The low aesthetical score of Tableau Public's Gallery is due to the images in this group being mostly infographics consisting of special glyphs and customized layouts that are not grids (e.g., Figure 10(h)). Such dashboard styles are of small number in the training data. Another reason for the low scores of LADV on the infographics is its weakness in terms of recovering large paragraphs of text. Our approach cannot accurately capture the area occupied by the text, along with the font and font size.

Most participants agreed that LADV could help them in their dashboard design despite the above-mentioned drawbacks Some comments are quite positive: *"I think I do not need to make any modification to some dashboards to implement the dashboard design into my project."* However, some participants suggested improving the recognition of text and chart details. One participant said, *"There are some font problems when I use LADV. The font detail will cause some changes in the dashboard perception."*. By contrast, another participant said, *"Some chart details are missing. I hope that this could be improved in the future."* We did not anticipate some participants indicating that the generated dashboard has better color than the original one. For example, one participant said (pointing at Figure 10(e)), *"The recognition result of the chart types is good. Sometimes, the automated color matching optimized the original dashboard. This is very surprising."*

The results show that LADV can help novices create dashboards. One participant said, *"I don't have much experience in dashboard design, but our company needs to update dashboard design frequently. This tool is definitely the one I was looking for. I can test different dashboard examples in a few minutes and choose the better one."*

7.3 User Study: LADV Workflow

We also conducted a user study to investigate the efficiency of the LADV workflow.

7.3.1 Study Design

We provided datasets and corresponding descriptions for presentation to motivate the participants to create dashboards with clear goals. However, the participants were not expected to import data on every attempt.

Creation tools. We compared LADV with a "basic" edition of DataV, which retains the interactive creation and customization of dashboards but removes recognition from images. The existing software was not chosen to ensure that the two tools have the same expressivity and usability in creating dashboards. Therefore, we implemented a tool that has similar UI and interactions to LADV to serve as a baseline for the controlled experiments.

Datasets. We prepared two multi-dimensional tabular datasets. The first one contained student information data of a university in Portugal, including student grades, family, social relationships, and other attributes. The dataset contains 649 instances and 33 attributes. The other dataset was from the US population census in 1994. This dataset contains 48,842 instances and 14 attributes; some examples of key attributes included work class, highest education, and capital

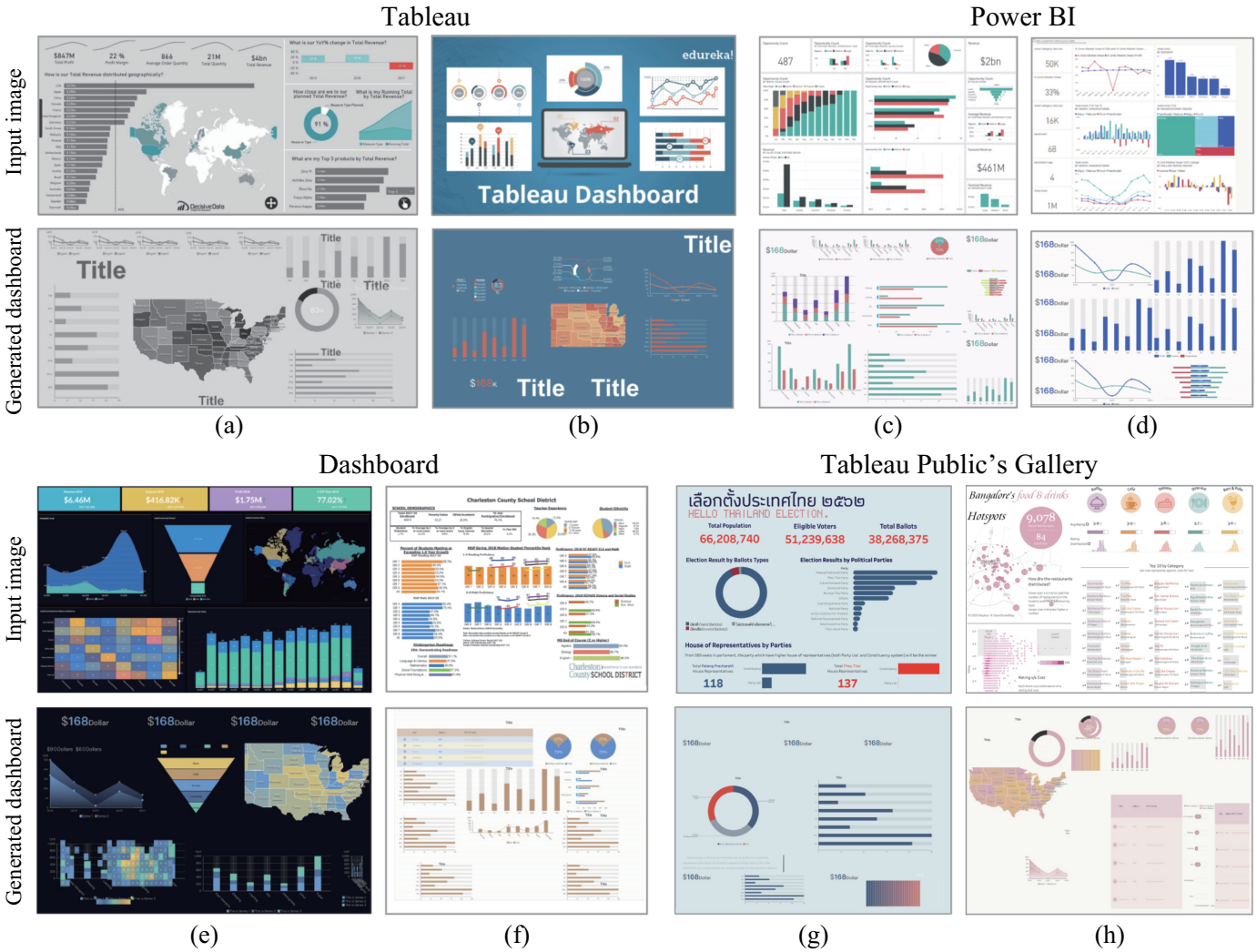


Fig. 10. Examples of dashboards learned and generated from images. The lower images are generated from the upper images in (a)–(h). The images come from Google image search results for the terms "Tableau", "Power BI", and "Dashboard", as well as Tableau Public's Gallery. (a), (c), (e), and (g) have the highest aesthetic scores in each group, and (b), (d), (f), and (h) have the lowest ones.

gain. However, the participants often spent a long time exploring the datasets during piloting. As we focused on the design decisions rather than data exploration, we prepared and provided collected patterns. The participants were asked to design their dashboards on the basis of the given information. For example, "There is a relationship between the number of school absences (numeric, from 0 to 93) and number of class failures (numeric, n if $1 \leq n \leq 3$, else 4)".

Each participant was assigned to two experimental conditions with different tools and different datasets to avoid the learning effect. The order of tools and datasets was counterbalanced across participants according to a Latin square design.

Participants and apparatus. We recruited 16 participants (88 females and 8 males, aged 24–35 years old, mean=27.3, STD=2.8). Half of the participants had used visualization tools[2], including DataV, Excel, Tableau, and Python/matplotlib. All experiments were conducted in a lab setting, with tools run in Chrome on a MacBook Pro with 13-inch Retina monitor, resolution of 2560 × 1600 pixels, and 60 Hz refresh rate. Each experimental condition lasted for about an hour. All participants were compensated with $8 per hour.

Procedure. The experiment for each condition began with a 10 min tutorial of the creation tool. The participants were shown an introduction to the tool's functionality, guide of interactions, and a sample of actual use (using a dataset different from the two utilized in the experiment). Thereafter, the participants were taught some dashboard examples with 20 images that could be browsed or imported to LADV during the entire experiment. Each participant could choose to sketch on paper, a whiteboard, or a tablet with a stylus. We also introduced the think-aloud method and asked the participants to follow. After the dataset and collected data patterns were introduced, the participants were asked to create dashboards. Specifically, the participants were asked, "Try to display the given information through different dashboard designs. Adjust the layout and style until you think the dashboard can properly demonstrate your design intention when communicating with others." The participants were also asked to mark the dashboards on which they felt that the design was completed. Subsequently, the participants were given 10 min each to create dashboards. However, the experiment could be terminated as long as the participants were satisfied with the accomplished dashboard(s).

2. 16 participants include 8 engineers (4 had used visualization tools, 4 has not), 4 designers (had not used visualization tools), and 4 college students (had used visualization tools).

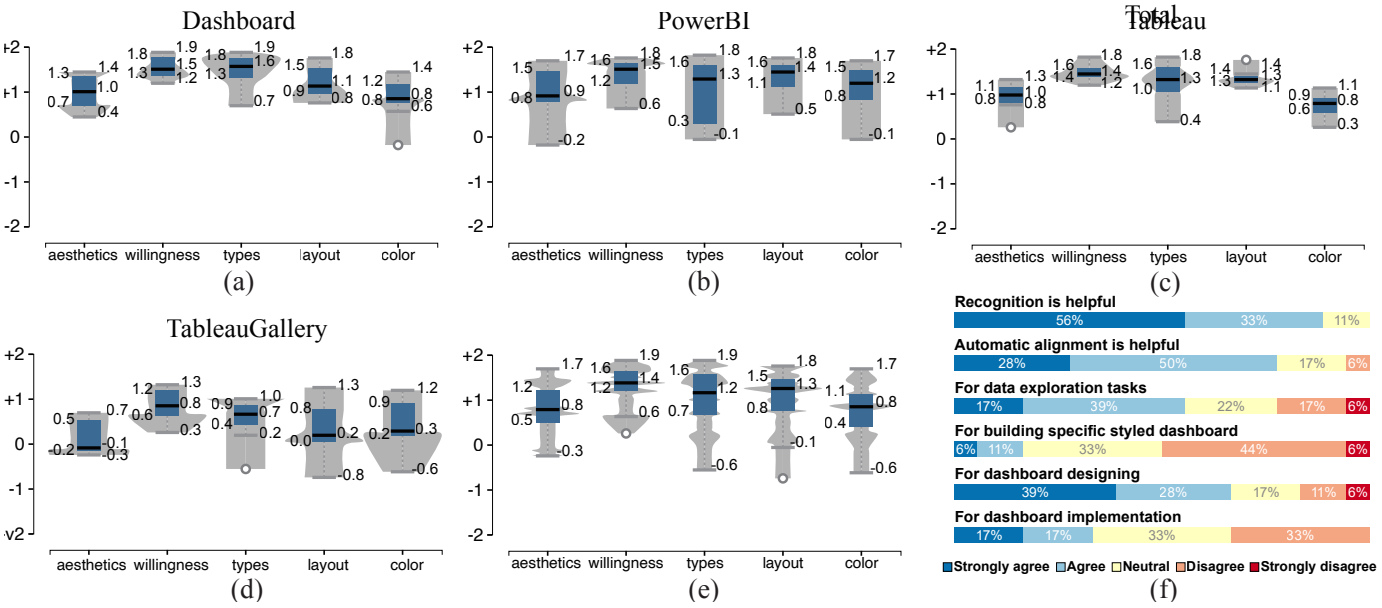


Fig. 11. (a–e) Distribution of participants' ratings on the generated dashboards from different collections of images. Participants rated aesthetics, willingness to continue to build upon the generated dashboard, types, layout, and color on a 5-point Likert scale (-2 strongly unsatisfied, -1 unsatisfied, 0 neutral, 1 satisfied, and 2 strongly satisfied). Ratings are grouped by different sources: Google image search for the term (a) "dashboard", (b) "Power BI", and (c) "Tableau", as well as (d) Tableau Public's Gallery, and (e) summary of all collected examples. (f) Results of the participants' ratings for LADV's ability to deal with different tasks on a five-point Likert scale.

Records and questionnaire. An experimenter observed the experimental process and took records. All interactions of the participants and resulting designs were automatically recorded, together with the audio to capture the participants' verbalizations. After each participant finished two experimental conditions, he/she was asked to fill out a questionnaire consisting of several comments and five-point Likert scale ratings.

7.3.2 Analysis and Results

We present some interesting results focusing on participants' workflow, efficiency, and experience using different tools.

LADV improves creativity. We analyzed the number of dashboard designs created to assess the mechanism by which LADV promotes creativity. Most experiments were terminated before the time limit was reached; thus, this number can reflect the participants' creative impulse. We found a significant difference between tools ($\chi^2(1,N=16)=6.0, p=0.014<0.05$). The participants create an average of 1.69 dashboards with LADV, whereas they created 1.19 dashboards with DataV (a 42% increase). This result indicates that LADV is beneficial for users in conceiving and attempting possible designs. User feedback also confirms this notion; for example, *"Compared with creating (charts) by drag and drop (in DataV), I continue to generate new ideas while hand-drawing (and then pass the sketches into LADV)."*

We then analyzed the difference between data analyzing novices and experienced participants. A significant difference in the number of dashboards created when using LADV was found between the two groups of participants ($\chi^2(1,N=16)=4.0, p=0.046<0.05$, novices: mean=1.25, others: mean=1.75). No significance was found when DataV is used ($\chi^2(1,N=16)=2.0, p=0.157<0.05$, novices: mean=1.38, others: mean=1.63). The results show that LADV can efficiently exploit users' understanding of the data.

Tools affect efficiency. We analyzed the effects of the tool on the time needed to finish creating a dashboard. The numbers of dashboards created by participants varied; thus, we only considered the first accomplished dashboard of each participant. On average, 205.9 s was needed to create a dashboard using LADV, and 356.9 s was needed for DataV. The time required for LADV was short, and a significant effect was observed ($\chi^2(1,N=16)=8.067, p=0.005<0.01$). We also observed a significant effect of the tool on the number of modifications made after creating charts and before being satisfied ($\chi^2(1,N=16)=12.25, p=0.00<0.01$, LADV: mean=1.06, DataV: mean=1.94). These results suggest that LADV can efficiently understand users' design intention and improve the efficiency of authoring dashboards.

Meanwhile, the dashboards created using DataV contained an average of 9.4 charts of 6.3 types, which are greater than the 8.1 charts of 5.4 types created with LADV. The possible reasons are as follows. LADV lacks hints for available chart type choices, and users often build up dashboards with the charts they are familiar with rather than try different types of charts.

Different workflows. We analyzed the effects of tools on the users' workflow. Only three of the participants (19% of the total 16 participants), who are all designers, attempted to generate dashboards from computer-generated images. One participant said, *"When I import a computer-generated dashboard image, I want the detailed style, but the resulting dashboard is different from what I expected."* This situation is attributed to the inability of LADV to identify detailed styles. Another participant noted that the computer-generated images lacked the advantages offered by sketching, which is more flexible. One designer stated that he is willing to generate dashboards by drawing detailed design drafts, which are more like computer-generated images rather than sketches. However, this task was not performed because it would have taken a substantial amount of time to complete during the experiment.

Participant feedback: usability and improvements. The participants' comments indicate that we achieved most of our goals, although some improvements can still be made. One

participant said, *"The learning curve of LADV is much lower and provides convenience. With LADV, the user can quickly setup the overall layout for further customization."* Another participant said, *"Automatic alignment is also very helpful."* Approximately 89% of the participants strongly agreed or agreed that learning styles from images is helpful, and the number for layout optimization is 78%. Figure 11(f) shows the participants' rating for LADV's ability to deal with different tasks. LADV is suitable for dashboard design, but it lacks support for data exploration and building specific dashboards. This finding is consistent with the comment stating that *"LADV is suitable for deciding the design and layout. DataV has low trial-and-error costs when designing the details."* Another similar comment indicated, *"I prefer LADV when conceiving different design ideas. DataV is more efficient to implement a given dashboard design."* Whether or not LADV is suitable as an implementation tool remains a controversial topic. Most participants agreed that the use of LADV in the early stage as a prototyping tool for dashboard creation is efficient. The results show that DataV can complement LADV. However, some participants noted that displaying the right data is more important than aesthetics, which LADV cannot cope with. For example, one participant said, *"The first thing I think about is not how the dashboard will look better, but how the information should be arranged. If possible, I hope the tool (LADV) can provide some layout recommendations."*

7.4 Review in Real Scenarios

LADV has already been deployed in the Alibaba Cloud computing environment and used in the actual design process of dashboard projects by PMs and designers. Feedback indicates that LADV is suitable for many usage scenarios. For example, one designer said, *"It (LADV) seems good when used for brainstorming."* Another designer suggests to sketch on erasable and reusable drawing medium, such as a white board. Figure 12 shows some designs that have been tried and evaluated after automatic generation under the real scenarios of Alibaba Cloud.

8 Discussions

LADV reshapes the process of creating a visualization dashboard. Rich designs with low workloads can be achieved by empowering users with the ability to conceive dashboards.

8.1 Template-Based vs. Free-Form Images

We first attempted to build a dashboard template framework that would provide appropriate dashboard designs on the basis of user requirements. However, the use of dashboard templates was eventually discarded because of two main problems. First, the variety of charts and layouts makes it impossible to cover all situations with a limited number of templates. Second, the template-based mode typically limits the imagination. By contrast, initialization from free-form images allows users in completing their own design intention rather than providing direct results.

However, our approach has its disadvantages. In contrast to template-based tools, LADV cannot guide users into trying different types of charts because they tend to only use charts in the galleries or those they are originally familiar with. A possible solution is to recommend alternative chart types.

8.2 Limitations

8.2.1 Chart Recognition

In testing our approach, we find some common chart recognition failure cases. Figure 13 shows some examples.

Most chart recognition failures of computer-generated images occur when the input dashboard contains a chart that is excluded from the training data. Figure 13(a)(d) show an example wherein a treemap is excluded in our training data and has been recognized as a basic heat map. Most of our training data come from the DataV visualization database. Accordingly, the recognition model does not perform as well in the infographics as in the regular dashboard design. Specifically, the regular trained chart features cannot cover these customized graphs. For example, LADV reached a low recognition result with infographic examples from Data-Driven [48] and InfoNice [49]. However, infographics mostly tend to focus on individual cases or a narrative story with specified graphs. By contrast, LADV is focused on expressing generalized interest.

Most chart recognition failures of sketched images are caused by irregular chart shapes, such as strokes that exceed the region of a chart, large circles in the scatterplot (Figure 13(b)(e)) or a bar chart with a small number of bars (Figure 13(c)(f)). Such mistakes also occur when the input image contains a control panel or other auxiliaries that are not charts (e.g., the top-right corner of the dashboard in Figure 14). Users' different drawing habits also cause failures. Sketches are drawn with many amenities such as grid lines of axes, often causing charts to be recognized as maps.

Another common problem comes from input images with only one chart. A single chart does not conform to the learned characteristics of dashboards. Such an issue can be easily eliminated by making a judgment before recognition.

8.2.2 Sophisticated Designs

Although LADV can recognize different chart types, it cannot learn from the design details, especially when the designer deliberately violates the visual design guidelines to achieve a special expression. For example, all line charts in Figure 14 share the same y-axis scale, resulting in the remaining three ones having seemingly incorrect aspect ratios. However, this design is to highlight the difference in value. The designer also added axis grids to eliminate the blank area, which may affect the overall aesthetics. Such sophisticated design leads to the incorrect judgment of LADV on the area occupied by the charts, resulting in a faulty blank area.

Nevertheless, as a preliminary attempt in this direction, the results of LADV are sufficient for initial conceptualization. The details can be later optimized by the rich customization options provided.

8.2.3 Usability for Novices

The generated dashboards are heavily influenced by user-supplied images. Although LADV can greatly alleviate the burden of designing and implementing dashboards, novices still have troubles in the beginning. Novices have neither accessibility to collect large quantities of dashboard examples nor the ability to judge the ones that are good. LADV provides dashboard collections for users to explore and select their preferred dashboard styles to solve this problem. LADV enables rapid prototyping for easy evaluation. Thus, novices can quickly try out different dashboard designs and find the one that they want most. Such rapid conceptualization process can provide a good opportunity for novices to learn from.

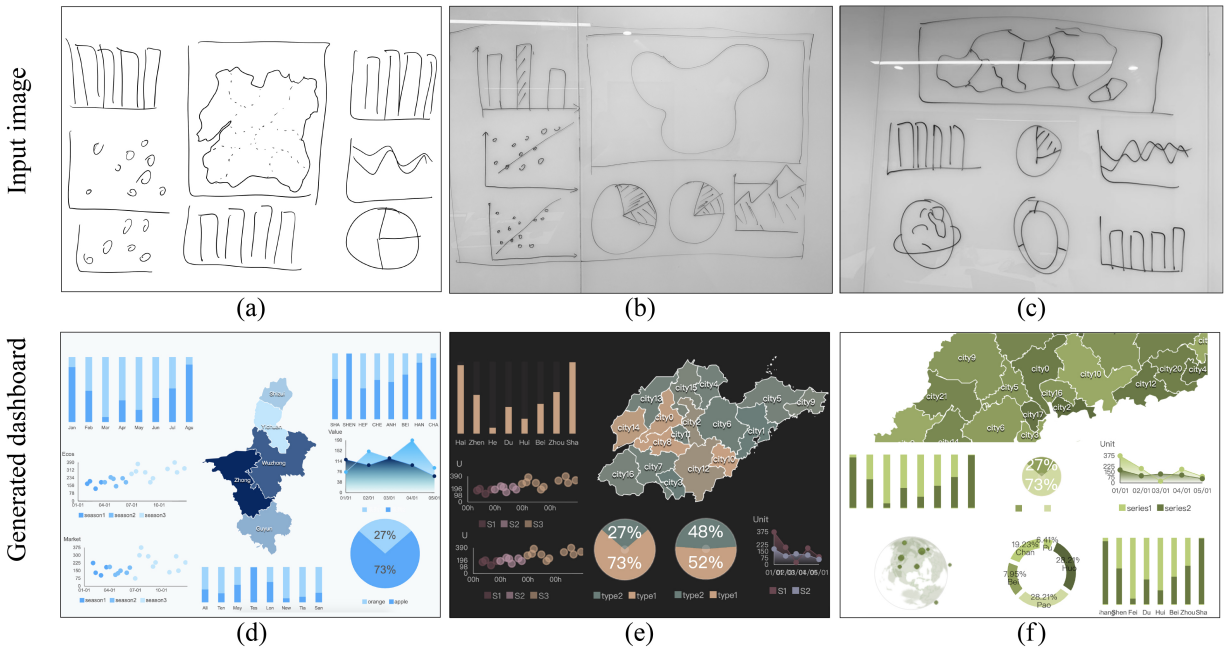


Fig. 12. Examples of dashboards learned and generated from sketches. (d–f) are generated from (a–c), respectively.

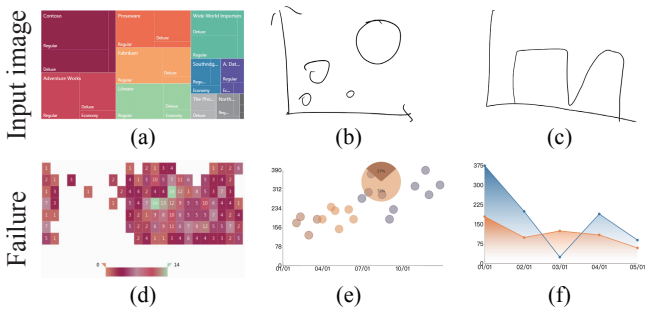


Fig. 13. Failure cases. (a) Treemap excluded from the training data and recognized as a similar chart (d). (b) A large circle in the scatterplot is recognized as (e) a pie chart. (c) A bar chart with small number of bars (<4) is recognized as (f) a line chart.

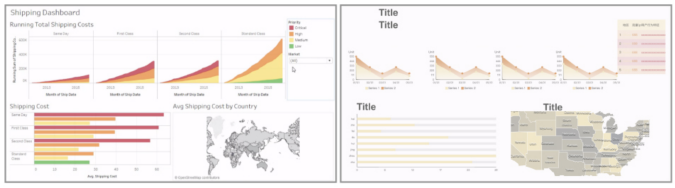


Fig. 14. LADV correctly recognizes the line charts but cannot understand the sophisticated design on the y-axis scale.

8.3 Future Improvements

In addition to further optimizing the model, we also summarized several potential improvements of LADV on the basis of the feedback from the user studies.

Data exploration and binding. Designing a good dashboard consists of three main aspects: understanding the data, design decisions for proper presentation of the data, and visual and aesthetical considerations. LADV is mainly for the second and third aspects as it allows users to leverage existing layouts or do thinking while sketching. The main advantage of LADV lies in its intuitive pipeline with minimal learning costs. However, the goal has not been fully achieved. At present, LADV fills the generated dashboard with a default dataset and lacks the recommendation for data dimensions and visual encodings. If LADV can recommend reasonable dimension binding on the basis of user-specified datasets and input exemplars, then it would be beneficial. This concept can also support the faceted browsing of datasets to help users efficiently discover data patterns. This idea is the main focus of our future work.

Detail style extraction. LADV efficiently performs in color extraction and layout optimization. However, detailed style extraction still requires improvement. For example, font is an important element in dashboard design, but it is not currently supported; only a default font is specified. Many participants of our user study indicated that font and other detailed styles, such as grid axis, legends, and text in the titles, are also worth learning from exemplars. Many successful methods for such extraction have been employed. We can easily integrate those methods into LADV to enhance its usability.

9 Conclusions

In this work, we propose LADV, which is a deep learning-based schema that applies chart inference to help users conceptualize their design intention. To the best of our knowledge, this study is the first attempt to automatically generate a visualization dashboard by using only images and sketches. LADV can promote the creativity of conceiving new dashboard designs. The results of our user studies show that LADV is suitable for dashboard designing tasks and can improve users' creativity.

Acknowledgments

We wish to thank all the anonymous reviewers for their valuable comments, and all the participants for their active participation. We also thank Ye Zhang from DataV Lab. The work is supported by the National Science & Technology Fundamental Resources Investigation Program of China (2018FY10090002), the National Natural Science Foundation of China (61772456, 61761136020, U1609217,61672538, 61872388, 61872389).

REFERENCES

[1] A. Sarikaya, M. Correll, L. Bartram, M. Tory, and D. Fisher, "What do we talk about when we talk about dashboards?" *IEEE transactions on visualization and computer graphics*, vol. 25, no. 1, pp. 682–692, 2019.

[2] H. Mei, Y. Ma, Y. Wei, and W. Chen, "The design space of construction tools for information visualization: A survey," *Journal of Visual Languages & Computing*, vol. 44, pp. 120–132, 2018.

[3] L. Grammel, M. Tory, and M.-A. Storey, "How information visualization novices construct visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 943–952, 2010.

[4] B. Lee, G. Smith, N. H. Riche, A. Karlson, and S. Carpendale, "SketchInsight: Natural Data Exploration on Interactive Whiteboards Leveraging Pen and Touch Interaction," in *Visualization Symposium (PacificVis), 2015 IEEE Pacific*. IEEE, 2015, pp. 199–206.

[5] N. W. Kim, E. Schweickart, Z. Liu, M. Dontcheva, W. Li, J. Popovic, and H. Pfister, "Data-Driven Guides: Supporting Expressive Design for Information Graphics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 491–500, 2017.

[6] Z. Liu, J. Thompson, A. Wilson, M. Dontcheva, J. Delorey, S. Grigg, B. Kerr, and J. Stasko, "Data illustrator: Augmenting vector design tools with lazy data binding for expressive visualization authoring," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 2018, p. 123.

[7] 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*. ACM, 2011, pp. 393–402.

[8] 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, 2018.

[9] L. Wilkinson, "The grammar of graphics," in *Handbook of Computational Statistics*. Springer, 2012, pp. 375–414.

[10] J.-D. Fekete, "The Infovis Toolkit," in *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*. IEEE, 2004, pp. 167–174.

[11] J. Heer, S. K. Card, and J. A. Landay, "Prefuse: a toolkit for interactive information visualization," in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2005, pp. 421–430.

[12] "Flare," http://flare.prefuse.org, accessed: March 2019.

[13] M. Bostock and J. Heer, "Protovis: A Graphical Toolkit for Visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 1121–1128, 2009.

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

[15] A. Satyanarayan, R. Russell, J. Hoffswell, and J. Heer, "Reactive vega: A streaming dataflow architecture for declarative interactive visualization," *IEEE Transactions on Visualization and Computer Graphics*, 2016. [Online]. Available: http://idl.cs.washington.edu/papers/reactive-vega-architecture

[16] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer, "Vega-lite: A grammar of interactive graphics," *IEEE Transactions on Visualization and Computer Graphics*, 2017. [Online]. Available: http://idl.cs.washington.edu/papers/vega-lite

[17] 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.

[18] C. Stolte, D. Tang, and P. Hanrahan, "Polaris: A system for query, analysis, and visualization of multidimensional relational databases," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 52–65, 2002.

[19] D. Ren, T. Höllerer, and X. Yuan, "iVisDesigner: Expressive Interactive Design of Information Visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2092–2101, 2014.

[20] H. Mei, W. Chen, Y. Ma, H. Guan, and W. Hu, "Viscomposer: A visual programmable composition environment for information visualization," *Visual Informatics*, vol. 2, no. 1, pp. 71–81, 2018.

[21] "Tableau Software," https://www.tableau.com, online; accessed Dec. 2019.

[22] "Power BI," https://powerbi.microsoft.com, online; accessed Dec. 2019.

[23] S. Few, "Information dashboard design," 2006.

[24] J. Browne, B. Lee, S. Carpendale, N. Riche, and T. Sherwood, "Data analysis on interactive whiteboards through sketch-based interaction," in *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. ACM, 2011, pp. 154–157.

[25] B. Lee, R. H. Kazi, and G. Smith, "Sketchstory: Telling more engaging stories with data through freeform sketching," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2416–2425, 2013.

[26] V. S. N. Prasad, B. Siddiquie, J. Golbeck, and L. S. Davis, "Classifying computer generated charts," in *2007 International Workshop on Content-Based Multimedia Indexing*. IEEE, 2007, pp. 85–92.

[27] J. Poco and J. Heer, "Reverse-engineering visualizations: Recovering visual encodings from chart images," in *Computer Graphics Forum*, vol. 36, no. 3. Wiley Online Library, 2017, pp. 353–363.

[28] D. Jung, W. Kim, H. Song, J.-i. Hwang, B. Lee, B. Kim, and J. Seo, "Chartsense: Interactive data extraction from chart images," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 2017, pp. 6706–6717.

[29] G. G. Méndez, M. A. Nacenta, and S. Vandenheste, "ivolver: Interactive visual language for visualization extraction and reconstruction," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 2016, pp. 4073–4085.

[30] P. Chagas, R. Akiyama, A. Meiguins, C. Santos, F. Saraiva, B. Meiguins, and J. Morais, "Evaluation of convolutional neural network architectures for chart image classification," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[31] 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.

[32] B. Tang, X. Liu, J. Lei, M. Song, D. Tao, S. Sun, and F. Dong, "Deepchart: Combining deep convolutional networks and deep belief networks in chart classification," *Signal Processing*, vol. 124, pp. 156–161, 2016.

[33] "Alibaba Cloud," https://www.alibabacloud.com/, online; accessed Dec. 2019.

[34] T. Munzner, "A nested model for visualization design and validation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 921–928, 2009.

[35] M. Rettig, "Prototyping for tiny fingers," *Communications of the ACM*, vol. 37, no. 4, pp. 21–27, 1994.

[36] J. C. Roberts, C. Headleand, and P. D. Ritsos, "Sketching designs using the five design-sheet methodology," *IEEE transactions on visualization and computer graphics*, vol. 22, no. 1, pp. 419–428, 2015.

[37] W. O. Chao, T. Munzner, and M. van de Panne, "Poster: Rapid pen-centric authoring of improvisational visualizations with napkinvis," *Posters Compendium InfoVis*, vol. 2, no. 1, p. 2, 2010.

[38] "7 tips and tricks from the dashboard experts," https://www.tableau.com/about/blog/2017/10/7-tips-and-tricks-dashboard-experts-76821, accessed: 2019-07-01.

[39] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[40] Z. Zhang, W. V. Stoecker, and R. H. Moss, "Border detection on digitized skin tumor images," *IEEE Transactions on Medical Imaging*, vol. 19, no. 11, pp. 1128–1143, 2000.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[42] M. Zhu, "Recall, precision and average precision," *Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, vol. 2, p. 30, 2004.

[43] K. Ferris and S. Zhang, "A framework for selecting and optimizing color scheme in web design," in *2016 49th Hawaii International Conference on System Sciences (HICSS)*. IEEE, 2016, pp. 532–541.

[44] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

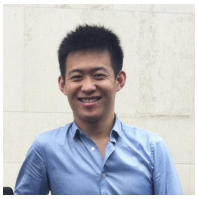
[45] S. Lok, S. Feiner, and G. Ngai, "Evaluation of visual balance for automated layout," in *Proceedings of the 9th international conference on Intelligent user interfaces*. ACM, 2004, pp. 101–108.

[46] T. Samara, *Making and Breaking the Grid, Updated and Expanded: A Graphic Design Layout Workshop*. Quarry Books Editions, 2017.

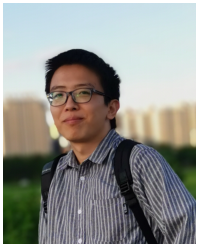
[47] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[48] N. W. Kim, E. Schweickart, Z. Liu, M. Dontcheva, W. Li, J. Popovic, and H. Pfister, "Data-driven guides: Supporting expressive design for information graphics," *IEEE transactions on visualization and computer graphics*, vol. 23, no. 1, pp. 491–500, 2016.

[49] Y. Wang, H. Zhang, H. Huang, X. Chen, Q. Yin, Z. Hou, D. Zhang, Q. Luo, and H. Qu, "Infonice: Easy creation of information graphics," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 2018, p. 335.

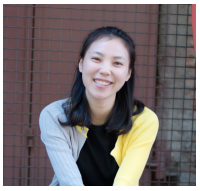


Ruixian Ma is a staff engineer at Alibaba Group. He received his B.A. degree at University of the Arts London, then he studied Information Experience Design at Royal College of Art, in 2016, he joined MIT Senseable City Lab as a research fellow. His research interests include machine learning driven approaches in information visualization and urban planning.

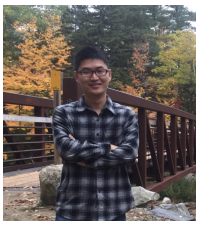


Honghui Mei received the Ph.D degree at 2019 at the State Key Lab of CAD & CG, Zhejiang University, China. He is now working in Alibaba Group, Hangzhou. His research interests include information visualization and visual analytics.

Huihua Guan received the B.S. degree at 2015 at Zhejiang University, China. She is now working in Alibaba Group, Hangzhou, China. She research interests include information visualization and visual analytics.



Wei Huang received the master degree at 2016 at Central South University, China. Her research interests include visual analytics and high dimensional visualization.



Fan Zhang is a postdoctoral fellow at MIT Senseable City Lab, he received his Ph.D. at The Chinese University of Hong Kong. His research interests is spatio-temporal data mining and social sensing.

Chengye Xin is a senior manager at Alibaba Group, She received his B.S. degree at Dalian University of Technology. Her research interests include information visualization and computing infrastructures.

Wenzhuo Dai is staff engineer at Alibaba Group, He received his B.S. degree at Anhui Polytechnic University. His research interests is computing infrastructures and database ststems.

Xiao Wen is a senior manager at Alibaba Group, He received his B.S. degree at Zhejiang A&F University. His research interests include information visualization and computing infrastructures.

Wei Chen is a professor at the State Key Lab of CAD & CG, Zhejiang University. Professor Chen received his PhD from the Zhejiang University in 2002. His research interests in- clude visualization, visual analytics, and biomedical image computing. For more information, please refer to http://www.cad.zju.edu.cn/home/chenwei.