SN Computer Science (2022) 3:500
https://doi.org/10.1007/s42979-022-01380-x

ORIGINAL RESEARCH

BarChartAnalyzer: Data Extraction and Summarization of Bar Charts from Images

Siri Chandana Daggubati[1] · Jaya Sreevalsan-Nair[1] · Komal Dadhich[1]

Received: 23 September 2021 / Accepted: 16 August 2022 / Published online: 1 October 2022

Abstract
Charts or scientific plots are widely used visualizations for efficient knowledge dissemination from datasets. However, these charts are predominantly available in image format. There are various scenarios where these images are interpreted in the absence of their source data table. This leads to a pertinent need for data extraction from an available chart image. We narrow down our scope to bar charts and its subtypes. We propose a semi-automated workflow, BarChartAnalyzer, for data extraction from chart images. Our workflow integrates the following tasks in sequence: chart type classification, image annotation, object detection, text detection and recognition, data table extraction, chart summarization, and, optionally, chart redesign. Our data extraction uses second-order tensor fields from tensor voting used in computer vision. Here, we propose a novel application of design study methodology for the chart summarization component. Our results show that our workflow can effectively and accurately extract data from images of different resolutions and subtypes of bar charts.

Keywords Chart classification · Chart segmentation · Chart image analysis · Optical character recognition · Data extraction · Text recognition · Text summarization · Convolutional neural Network · Design study methodology · Bar charts · Stacked bar charts · Grouped bar charts · Histograms

Introduction

Data can be interpreted better when presented as visualizations, wherein one of the simplest and most ubiquitous forms is the class of charts. Chart representation specifically is a widely used approach, which is evident from the inclusion of the basic understanding of simple charts in the curriculum of primary school education. Simple charts, e.g., bar charts, scatter plots, etc., are commonly found in documents (textbooks, publications), print media (newspapers, magazines), and on the Internet; and are most prevalent in image format. There are use cases of redesign and reconstruction of charts for getting high-resolution images for applications such as generating accessible reading materials for differently abled students. The chart redesign also enables students with learning difficulties to understand data using alternative designs. These applications pose a problem when the source data for the charts are not available alongside the chart image for ready consumption. Thus, data extraction in the form of semi-structured tables [1] from these chart images is a relevant problem, specifically in the space of improving assistive technologies.

Chart interpretation includes data extraction and textual summary generation. The redesigning of multi-class charts is a motivating application of the data extraction, as they are relatively difficult to interpret [2]. The redesign entails the requirement of source data that is used to generate the original plot as well as information about multiple classes being represented in the image. A motivating application of chart summarization is for generating alternative text (alt

S. C. Daggubati, J. Sreevalsan-Nair, and K. Dadhich have contributed equally to this work.

This article is part of the topical collection "Image Processing and Vision Engineering" guest edited by Sebastiano Battiato, Francisco Imai and Cosimo Distante.

✉ Jaya Sreevalsan-Nair
jnair@iiitb.ac.in

Siri Chandana Daggubati
daggubati.sirichandana@iiitb.ac.in

Komal Dadhich
komal.dadhich@iiitb.org

1 Graphics-Visualization-Computing Lab, IIIT Bangalore, 26/C Electronics City, Hosur Road, Bangalore, Karnataka 560100, India

text) for web and document accessibility. While there are applications where user intent can refine the chart summary [3], use of the extracted data facilitates more generic applications where the goal is to inform the user of the source data of the chart. In applications pertaining to accessibility, chart summaries succinctly describe the data better than the data table. One of the approaches for generating the textual summary of chart images generically and yet automatically is using its extracted data along with its extracted textual content.

Since the design space for charts is large, in terms of chart types and their formatting, we focus on a single chart type here. Amongst all statistical plots, bar chart representation is the most commonly used one for visual summarization. Bar charts have subtypes, depending on the data type and user requirement, such as simple, stacked, grouped bar charts, to name a few. Stacked and grouped bar charts help visualize multi-class or multi-series data. The grouped bar chart gives inter-and intra-class trends, and the stacked bars give part-to-whole information for multiple classes. Overall, we focus on bar charts and its subtypes here.

The state-of-the-art reasoning over scientific plots includes bar charts as a chart type of interest [1, 4]. These methods use convolutional neural networks (CNNs) for the object detection component for extracting bar geometry. However, we find that CNNs for bar extraction may fail specifically for the stacked bars. Hence, we use the image processing method exploiting spatial locality for object detection [5]. We, thus, propose a semi-automated workflow (Fig. 1), called BarChartAnalyzer [6], that can take an image as input, identify the bar chart, sub-classify it to bar chart type, and then perform data extraction. The extracted data can be further used for reconstruction to get customized charts of high-resolution image quality, as well as for redesigning the complex to simpler charts. Here, we extend our previous work on BarChartAnalyzer by improving the text summarization component using a novel design study. Our approach is inspired by the design study methodology (DSM) used for generating data visualizations and information graphics [7]. In summary, our contributions are:

- an end-to-end semi-automated workflow (Fig. 1) for interpreting images of bar charts and their seven subtypes, with an improved text summary,
- a novel design study-based approach for template-based chart summarization for bar chart subtypes,
- training dataset for bar chart images for bar chart subtype classification, along with text summaries.

Our most significant contribution here is the text summary generating component (C7), for which we adopt DSM, which is usually used for visualization design. Here, we propose the text summary generation from chart images as its novel application. We have designed and implemented a user study as a part of DSM. Our current work builds on our previous work of the use of tensor fields for object (i.e., bar) detection [5], and the proposed workflow (C1–C7) in BarChartAnalyzer [6]. In the latter, chart summarization relies entirely on a template, which is now improved here by the observations from a user study.

Related Work

Chart interpretation is generally divided into smaller tasks such as chart type classification, data extraction, and, optionally, reconstruction or redesign, and summarization. ReVision is a system that performs tasks like identifying chart

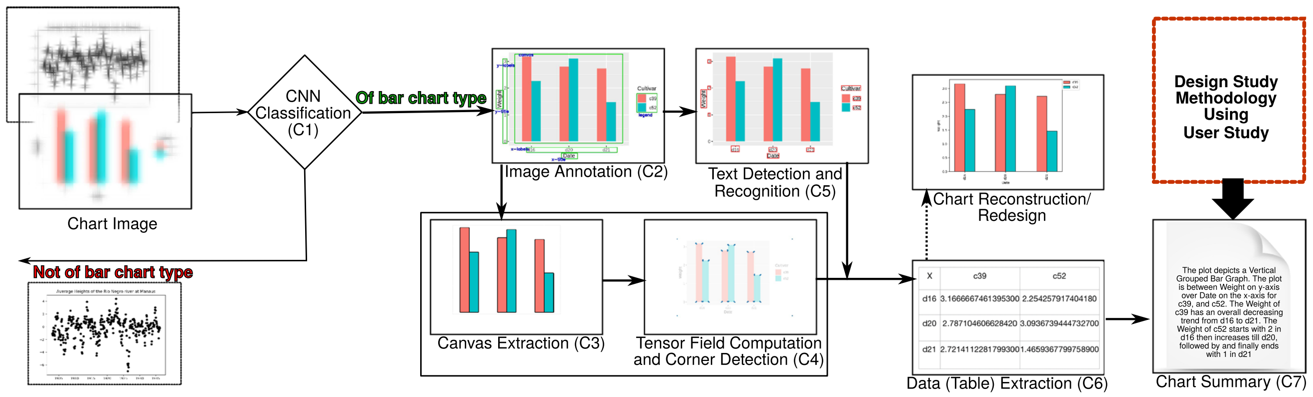


Chart Image
CNN Classification (C1)
Of bar chart type
Not of bar chart type
Image Annotation (C2)
Text Detection and Recognition (C5)
Chart Reconstruction/ Redesign
Design Study Methodology Using User Study
Canvas Extraction (C3)
Tensor Field Computation and Corner Detection (C4)
Data (Table) Extraction (C6)
Chart Summary (C7)

X	c39	c52
d16	3.1666667461395300	2.254257917404180
d20	2.787104606628420	3.0936739444732700
d21	2.7214112281799300	1.4659367799758900

The plot depicts a Vertical Grouped Bar Graph. The plot is between Weight on y-axis over Date on the x-axis for c39, and c52. The Weight of c39 has an overall decreasing trend from d16 to d21. The Weight of c52 starts with 2 in d16 then increases till d20, followed by and finally ends with 1 in d21

Fig. 1 Our proposed workflow for data extraction from a given chart image using our proposed semi-automated BarChartAnalyzer (BCA), with seven components (C1–C7), for applications including chart reconstruction and redesign. Significant components are C1 for classifying the chart image to bar charts and its subtypes, C2–C4 for feature extraction, C5 for text detection for data contextualization, and C6 for data table generation. Here, we use the design study methodology for improving the text summary in C7. This image is a modified version of the one in our previous work [6]

type, extracting visual elements, and encoded data by creating feature vectors and identifying geometric structures in pixel space [8]. WebPlotDigitizer is another system that provides both automatic and manual procedures to extract data from given chart images [9]. However, the tool requires extensive user interaction for aligning axes to select data points. It works for simple bars, but fails for stacked and grouped bar charts in giving class information.

Machine learning models have been effectively used for classification and/or object detection problems in chart analysis. Beagle is a web-based system for classifying charts in scalable vector graphics format [10]. Text type classification has been done using feature vector generated using the geometric property of text along with mark type classification using a fine-tuned AlexNet [11]. FigureSeer uses a similar fine-tuning approach [12]. A convolutional neural network (CNN) model used for chart classification can also be used for object detection, e.g., for chart objects such as bars in the source image [4]. ChartSense uses GoogleNet to classify line, bar, pie, scatter charts, map, and table types [13]. ChartSense further uses the connected components method to extract bar objects, using the x-axis as a baseline in the image. While this method works for simple bar charts, the charts with bars of multiple classes (e.g., grouped bars) get incorrectly identified as belonging to the same class. The existing methods using object detection-based approach have not been shown to work for all subtypes of bar charts, e.g., stacked bars, for which training data is currently unavailable. Hence, we rely on conventional computer vision and image processing methods on all bar chart subtypes to identify bars, and bar segments, in the case of stacked bars. We choose a corner detection method that exploits spatial locality using second-order tensor fields [5] in BarChartAnalyzer. These methods largely fall under the category of chart interpretation through visual structure extraction.

Apart from visual structure extraction, chart analysis can be done through automated chart question–answering (CQA) systems, and data parsing. PlotQA is an example of CQA that uses a more accurate neural network for object detection for visual elements, such as bars [1]. While PlotQA is an example of an approach where data are parsed from charts, and then used for QA, an alternative approach for CQA is using natural language understanding (NLU) methods. An example of such an approach is through the use of transformers for answering questions from charts directly, of which Structure-based Transformer using Localization, STL-CQA [14], is a recent solution.

Text detection is important for chart inference. Automated data extraction for bar charts has been done by identifying graphical components and text regions independently [15, 16]. Text detection and recognition are both required for extracting the text content from a chart image. Chart data are finally extracted using inference from both graphical and text content.

A textual summary of a chart is a relevant task for its interpretation. While its relevance may appear counter-intuitive as the charts are themselves visual summaries of data, its text summarization has been found to be useful for the visually impaired (VI) users to read its images, usually embedded in documents. Visualization of summaries in a graphical interface is an alternative to text summaries of charts, e.g., Chartseer [17], but does not improve the accessibility to the VI users. Hence, the conventional textual summaries are of interest to us. The challenges involved in textual summarization are in the selection of relevant information, organization of the content into fluent text, use of appropriate comprehensible sentence structures, and choice of appropriate expressions in the concerned language, i.e., English in this case [18]. The purpose of the chart summary determines the relevant information that is added to the content [16]. Once the content is identified, the summary is generated using two different natural language generation (NLG) approaches, namely, predefined templates and deep-learning methods. The predominantly used template-based approach provides more rigid content than the deep learning methods that is more recently used. For generic applications or for creating master summaries, the former is simpler to implement and also suffices.

Using predefined templates, iGRAPH-Lite system generates a short summary explaining the visual description of the chart itself, but does not provide insight into the information provided by visualization using the chart [19]. Linguistic constructs have been used to generate a chart summary with the help of semantic graph representation [20]. Three types of features have been selected from charts, namely, salience, trend, and rank, that encode details like increasing or decreasing trend, any specific colored bar showing highly prominent detail. However, we have found that the summary output of this system has limited description. Summaries, especially for bar charts, have also been generated by calculating differences using existing attributes in chart images and providing the core message represented by the selected chart [21]. In an improved version, the level of importance of the different aspects of the content has been determined using a user study [18]. A centrality-based algorithm based on PageRank has been used for content selection prior to templatized sentence creation for the bar (simple and grouped) and line charts for improving accessibility of information graphics to the VI users [22]. Unlike these methods, we use the design study methodology used in human–computer interaction (HCI) [7] as an alternative for content selection. Our work is also different from users collaboratively designing chart summaries using a graphical interface, e.g., Chart Constellations [23].

Recently, deep-learning methods in NLG have been used for chart summarization. These include the 1D convolutional residual network in Autocaption for creating captions of visualizations [24], long- and short-term memory network (LSTM) with user inputs in cyberphysical social systems [3], and transformer-based encoder–decoder architecture in Chart-to-Text [25].

The 7-Component Workflow

We propose a workflow, BarChartAnalyzer, that interprets a given chart from its image. It has seven main components (Fig. 1), namely, chart subtype classification, chart image annotation, canvas extraction, tensor field computation, text recognition, data table extraction, and chart summarization.

Chart Subtype Classification (C1): Data are represented using different chart types based on the number of variables and user requirements. For example, scatter plots and bar chart representations visually encode the data differently, requiring different chart analysis approaches. In the case of bar charts, there are commonly used subtypes, namely, simple bars, grouped bars, stacked bars, and of different orientations, depending on design requirements. Since the data extraction process from a given chart image depends on its chart type, identifying the type/subtype is the first step of our workflow.

Image classification is a widely studied problem in computer vision, and it has been done using different CNN-based classification models, such as AlexNet and GoogleNet. These models were trained and tested for a set of natural images provided during the ImageNet challenge [26]. The natural images contain characteristics other than the shape of the object, like texture, finer edges, color gradients, etc. However, compared to natural images, the chart images are sparser and more structured with repeating patterns. Hence, the models that work for natural images do not work effectively for chart images.

The chart objects, such as bars, scatter points, and lines, are distinguishable based on their shape and geometry, unlike objects found in natural images. However, the chart subtypes for bar charts all have similar geometry, i.e., bars. Hence, contour-based techniques for chart subtype classification are inefficient. A few pre-trained models have been used for chart type classification of images by imposing certain constraints, e.g., training with a small image corpus; however, the classification outcomes have low accuracy. The classifier in ChartSense has used GoogleNet [13], which has been trained on different chart types. This model classifies subtypes, such as grouped and simple bars, also, since the features are similar in both subtypes. However, other subtypes of our interest, namely stacked bar charts, have not been explored. On the other hand, mark-based chart classification is an alternative approach [11], where the classifier is trained to recognize five mark types: bars, lines, areas, scatter plot symbols, and other types. This classifier is limited to the identification of the chart type without any scope of extension to subtypes. Thus, we explore all bar chart subtypes, including the stacked bar charts and histograms, that have not been considered widely in the state-of-the-art.

Our proposed subtype classifier is inspired by the VGGNet (Visual Geometry Group Network) architecture [27], which is widely used for object detection and segmentation tasks on image databases, such as KITTI [28], an image benchmark dataset for road-area, and ego-lane detection [29]. We choose the VGGNet architecture as it is efficient in feature extraction from images and addresses depth in convolutional networks. It is also simple for a new model, with the flexibility of adding more VGG blocks. The VGGNet architecture has a stack of convolutional layers (Fig. 2 (left)) that generalizes the deep-learning tasks. Our CNN model is a combination of the convolutional, pooling, and fully connected layers, as specified in Table 1. The convolutional layers are responsible for extracting features by convolving images using kernels. Our classifier uses max-pooling to reduce computation by reducing the spatial size by half. The tailing layers in our classifier are the fully connected layers that take the results of the pooling/convolutional layer and assign it a label/class. Our classifier identifies the bar chart subtype of an input image. This classification also checks if the given image is of bar chart type, as the workflow downstream accepts only bar charts, rejecting the others.

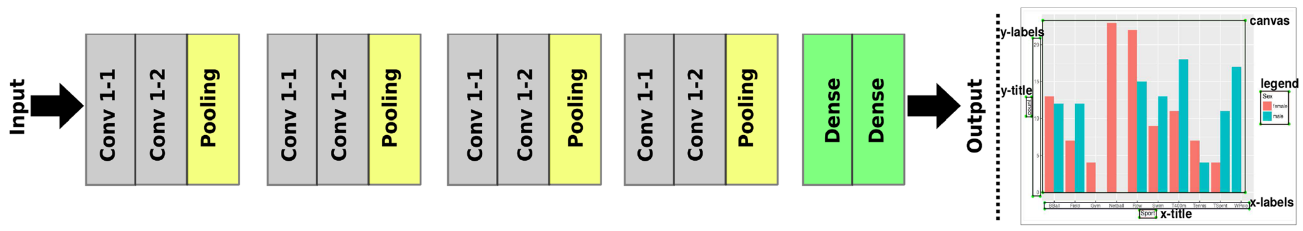


Fig. 2 (*Left*) The architecture diagram of our CNN-based classifier for identifying bar chart subtypes. (*Right*) Human-guided annotation of the chart image, where the chart canvas is used for object detection. This image is a modified version of the one provided in our previous work [6], specifically for the CNN architecture diagram

Table 1 The VGGNet [27]-inspired architecture of our proposed convolutional neural network for chart type classification, diagrammatically given in Fig. 2

Layer	Type	Output shape
input	_	$200 \times 200 \times 3$
conv1	Convolution	$200 \times 200 \times 16$
conv2	Convolution	$200 \times 200 \times 16$
pool1	Max-pooling	$100 \times 100 \times 16$
conv3	Convolution	$100 \times 100 \times 16$
conv4	Convolution	$100 \times 100 \times 16$
pool2	Max-pooling	$50 \times 50 \times 16$
conv5	Convolution	$50 \times 50 \times 32$
conv6	Convolution	$50 \times 50 \times 32$
pool3	Max-pooling	$25 \times 25 \times 32$
conv7	Convolution	$25 \times 25 \times 64$
conv8	Convolution	$\times 25 \times 64$
pool4	Max-pooling	$12 \times 12 \times 64$
fc1	Fully connected	256
fc2	Fully connected	8

Training Dataset—The chart image dataset that we have curated consists of images of seven different subtypes of bar charts, namely, simple, grouped/clustered, and stacked bar charts of horizontal and vertical orientations and histograms. The training dataset of our CNN model consists of images of these seven subtypes and additionally an "other" category. The "other" category includes scatter plots and line and pie chart images that are commonly used, excluding bar chart subtypes. Histograms are included as one of the subtypes of bar charts, since some plotting tools, e.g., Google sheets and Microsoft Excel®, use bar charts for histogram plots. Also, we observe that the geometry of bins in histograms is the same as columns/bars in the bar charts. Our CNN-based classifier requires input images of fixed size for training; hence, we first resize the images in the dataset to 200×200 size. The image resizing and classifier implementation has been done using Python imaging (`PIL`) and `Keras` libraries, respectively. Our CNN model for chart subtype classification is novel in its application for classifying bar chart subtypes. Our classifier assigns class labels to an input image specifying the bar chart subtype and its orientation, except in the case of histograms, e.g., "horizontal grouped bar," "vertical stacked bar."

Image Annotation (C2) and Canvas Extraction (C3): Image annotation is usually performed to generate training datasets for computer vision-related problems like object detection, segmentation, etc. Image annotation entails labeling different regions of interest (ROIs) in the images. These predefined labels are used to detect and extract ROIs in test data using learning approaches. Since this requires contextual labels and appropriate associations between labels and ROIs, human-guided annotation is an optimal approach.

For chart images, manual marking and annotation of bounding boxes for ROIs have been widely used [1, 4]. Different labels are assigned to the components of chart images based on their role in the visualization, such as canvas, x-axis, y-axis, x-labels, y-labels, legend, title, x-title, and y-title, as shown for a sample chart image [Fig. 2 (right)]. We use LabelImg [30] to mark and annotate bounding boxes for ROIs of the above-mentioned labeled components of a chart image. `LabelImg` is a Python tool with a graphical user interface (GUI) for interactively selecting an image, drawing a bounding box for an ROI, annotating the ROI, and labeling the ROI. We use the label *Canvas* for the ROI that contains the chart objects, such as bars, lines, or scatter points, and is defined as *chart canvas*, which is one of the chart image components [5]. The annotation is generated as an XML file that is processed to extract the canvas region as well as for text localization. The former is used for chart extraction (C3), and the latter for text detection (C5).

The canvas extraction step (C3) includes image preprocessing methods to remove the remaining elements other than chart objects such as gridlines, overlaid legends, etc. The subsequent step (C4) on tensor field computation is sensitive to the presence of these extraneous elements, which leads to erroneous results. Image processing techniques, marker-based watershed segmentation, and contour detection algorithm have been used to remove such components in the chart canvas effectively [5]. These steps also fill hollow bars, as required, since the tensor field is computed effectively for filled bars. Highly pixelated edges in aliased images lead to uneven edges in each bar object. This issue is addressed using the contour detection method to add a fixed-width border to bars [5]. Overall, we perform these steps, as illustrated in Fig. 3 with an example, to extract a chart canvas containing chart objects used in C4.

Tensor Field Computation (C4): Tensor fields have been widely used to exploit geometric properties of objects in natural images [31] using structure tensor and tensor voting. We use a local geometric descriptor as a second-order tensor for tensor vote computation [32] that further leads to corner detection in the case of bars for a given bar chart. The steps in C4 are illustrated in Fig. 4 with the help of an example.

Structure tensor T_s at a pixel provides the orientation of the color gradient computed from the local neighborhood. T_s is the gradient tensor T_g at the pixel with intensity I; convolved (using $*$ operator) with Gaussian function $\mathcal{G}$ with zero mean and standard deviation ρ. Then, T_s is given as follows:

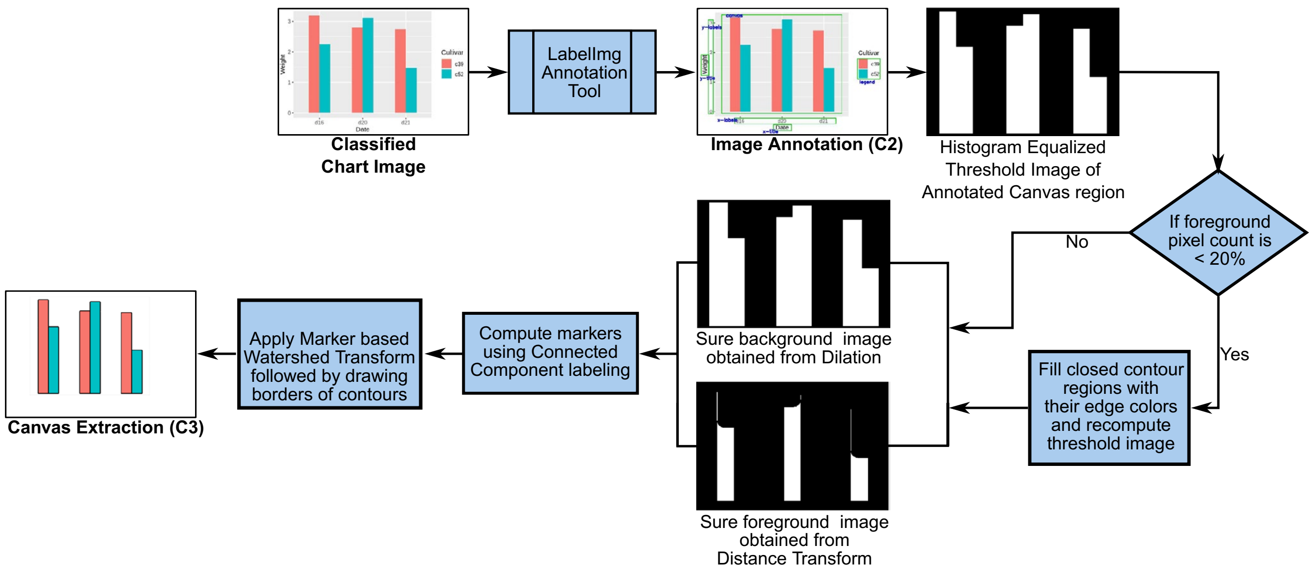

Classified Chart Image
LabelImg Annotation Tool
Image Annotation (C2)
Histogram Equalized Threshold Image of Annotated Canvas region
If foreground pixel count is < 20%
No
Yes
Fill closed contour regions with their edge colors and recompute threshold image
Sure background image obtained from Dilation
Sure foreground image obtained from Distance Transform
Compute markers using Connected Component labeling
Apply Marker based Watershed Transform followed by drawing borders of contours
Canvas Extraction (C3)

Fig. 3 The illustration of the workflow for C2 and C3 demonstrates the chart image annotation and canvas extraction sub-components of the BarChartAnalyzer with their intermediate steps. Here, we follow the steps in C2-C3 as described in our previous work [6]

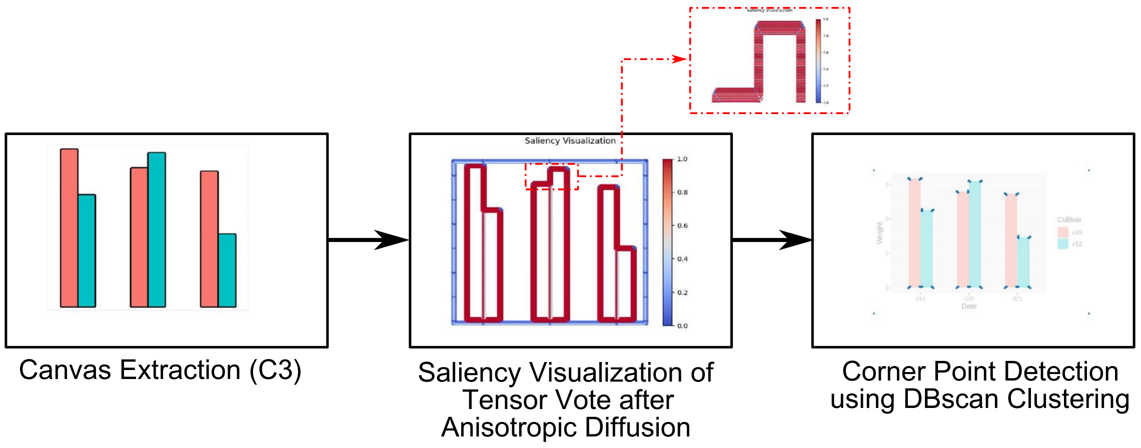

Saliency Visualization
Canvas Extraction (C3)
Saliency Visualization of Tensor Vote after Anisotropic Diffusion
Corner Point Detection using DBscan Clustering

Fig. 4 Illustration of the steps in C4 for corner point extraction of bar chart image using tensor field computation using DBscan clustering of critical points from anisotropically diffused tensor voting with saliency value $C_l < 0.4$. Here, we follow the steps in C4 as described in our previous work [6]

$$T_g = (G^T G), \text{ where gradient vector } G = \begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix}, \\ T_s = \mathcal{G}_\rho * T_g. \tag{1}$$

The tensor vote cast at x_i by x_j using a second-order tensor K_j in d-dimensional space is S_{ij}, computing using the closed-form Eq. [33]. If d_{ij} is the distance vector, i.e., $d_{ij} = x_j - x_i$, then its unit vector is $r_{ij} = \hat{d}_{ij}$. Let I_d be the d-dimensional identity matrix, and σ_d be the scale parameter which is used with the inverse distance weight using Gaussian function, $c_{ij} = \exp\left(-\frac{\|d_{ij}\|_2^2}{\sigma_d}\right)$. Thus, we compute S_{ij} as

$$S_{ij} = c_{ij} R_{ij} K_j R'_{ij}, \\ \text{where we compute } R_{ij} = \left(I_d - 2 r_{ij} r_{ij}^T\right); \\ R'_{ij} = \left(I_d - \frac{1}{2} r_{ij} r_{ij}^T\right) R_{ij}. \tag{2}$$

Here, we use the gradient tensor T_g as K_j [34]. If $r_{ij} r_{ij}^T$ is the distance tensor, then R_{ij} is physically the normal tensor to the distance tensor, when treated as the tangent tensor.

We compute the color gradient using the color in the CieLAB space [5]. Owing to the use of color for corner detection, BarChartAnalyzer fails for bar charts that use texture for region-fill of the bars. This is observed especially in the case of grouped and stacked bar charts where texture is used to differentiate bars of different classes/series.

Anisotropic diffusion As the tensor votes T_v in normal space has to encode object geometry in tangential space, we perform anisotropic diffusion to transform T_v to tangential space [5, 32]. The eigenvalue decomposition of the two-dimensional T_v yields ordered eigenvalues, $\lambda_0 \geq \lambda_1$, and corresponding eigenvectors v_0 and v_1, respectively. Anisotropic diffusion requires a *diffusion parameter* δ, for which ($\delta = 0.16$) is widely used [5, 35]. Anisotropic diffusion of T_v gives $T_{\text{v-ad}}$, that is a positive semidefinite second-order tensor

$$T_{\text{v-ad}} = \sum_{k=0}^{1} \lambda'_k . v_k v_k^T, \quad \text{where} \ \ \lambda'_k = \exp\left(-\frac{\lambda_k}{\delta}\right). \tag{3}$$

Saliency computation The saliency of a pixel to belong to geometry features of line- or junction/point-type is determined by the eigenvalues of $T_{\text{v-ad}}$ [5]. We get the saliency maps at each pixel of an image of its likelihood for being a line- or junction-type feature, C_l and C_p, respectively

$$C_l = \frac{\lambda_0 - \lambda_1}{\lambda_0 + \lambda_1} \ \ \text{and} \ \ C_p = \frac{2\lambda_1}{\lambda_0 + \lambda_1}, \tag{4}$$

using eigenvalues of $T_{\text{v-ad}}$ of the pixel, such that $\lambda_0 \geq \lambda_1$. The pixel with $C_p \approx 1.0$ is referred to as a critical point or degenerate point in the parlance of tensor fields. We detect all the critical points in the chart canvas during C4.

DBSCAN clustering The critical points of the chart image computed from tensor field computation form sparse clusters at the corners of each bar [5]. These pixels are localized using density-based clustering, DBSCAN [36], and cluster centroids are computed by tuning the hyperparameters of DBSCAN clustering to specific chart types. These cluster centroids are treated as corners of the bar. Using the positional layout or arrangement of these corner points based on the specific chart type and subtype, we heuristically compute the height of each bar in pixel space.

Text Recognition (C5) and Information Aggregation for Data Extraction (C6): At this juncture, the data we have extracted from the chart image, using tensor field computation, are in the image (or pixel) space. However, the extracted data must be in the data space for accurately summarizing and optionally reconstructing the chart. Hence, to transform the data from the pixel space to the data space, we now combine the data in pixel space with the text information in the image. We perform text detection to get x-axis and y-axis labels and compute the scale factor between the pixel and data spaces. The recognition of other textual elements, namely, plot title, legend, x-axis, and y-axis titles, also plays a crucial role in analyzing chart images, e.g., the information is used in summary (C7).

The deep-learning-based OCR, namely Character Region Awareness for Text Detection, CRAFT [37], is used for effective text area detection, including arbitrarily-oriented text. This approach is designed for relatively complex text in images, and it works by exploring each character region and considering the affinity between characters. A CNN designed in a weakly supervised manner predicts the character region score map and the affinity score map of the image. The character region score is used to localize individual characters and affinity scores to group each character to a single instance. Therefore, the instance of text detected is not affected by its orientation and size. The text orientation is inferred from the detected text boxes and then rotated to horizontal orientation for the proper extraction.

The CRAFT text detection model can be followed by a unified framework for scene text recognition that fits all variants of scenes, called the scene text recognition framework STR [38]. Being a four-stage framework consisting of transformation, feature extraction, sequence modeling, and prediction, STR resembles the combination of computer vision tasks such as object detection and sequence prediction task and, hence, uses a convolutional recurrent neural network (CRNN) for text recognition. We find that the CRAFT model, along with the STR framework, works efficiently to retrieve labels and titles of the chart image better than the widely used Tesseract OCR [39]. This is because Tesseract OCR fails for commonly found characteristics of chart images, such as text content with different colors, sizes, orientations, curvy fonts, and different languages, along with interferences or issues in the text, such as low resolution, exposure, noise, motion blur, out-of-focus, varying illumination, etc. Thus, we use the CRAFT model with the STR framework here.

In C6, subsequent to text detection and recognition, we transform the data extracted in pixel space to data space and add appropriate textual information for the variable name and bar width. Once the information is in the data space, we extract the data table. For both stacked and grouped bar charts, we additionally identify class/series information using the legend. The class/series information is then added to the extracted data table.

Chart summarization (C7) Our next step is to generate a summary of the chart image based on its retrieved data table and the perceivable visual elements of the chart image. While our approach is to use a predefined template for generating the textual summary, we propose a design study for generating it, as explained in section "Design Study for Chart Summarization". A design study is apt here, since this is a task involving human–computer interaction (HCI), and the output of the workflow, i.e., the summary, is to be *designed* for real users [7]. The design study is necessary to facilitate consideration of the user inputs during summary generation. This is different from the conventional practice of using user study for evaluation of chart summary [3, 25]. Thus, in our work, we utilize a user study for design iterations of the summary template.

Design Study for Chart Summarization

An effective summary should concisely represent the significant information of the chart image instead of a detailed textual representation of the entire chart data table. The information perceived as significant by a user varies based on various aspects, such as education, work, and their reasoning. Given our requirement of creating a generic summary without serving any specific intent, a predefined template suits our requirement better than the deep learning solutions based on user inputs [3].

We use the 9-stage design study methodology [7] to arrive at the predefined template to be used for the chart summary. Even though the design study methodology pertains to visualization as an output of human–computer interaction (HCI), it can also be extended to the *design* of chart summaries. This is because the chart summary is computer-generated but consumed by human users, just like visualizations themselves. The nine stages are divided into three top-level phases in sequence, namely *precondition*, *core*, and *analysis*, and the nine stages are split across these phases. The precondition phase, which focuses on the design's preparation, includes *learning*, *winnow*, and *cast* stages in sequence. The core phase, central to the actual design process, includes *discover*, *design*, *implement*, and *deploy* stages in sequence. The analysis phase, which is done retrospectively, includes *reflect* and *write* stages in sequence. Even though these nine stages form a linear process, the design study involves several feedback loops leading to a highly iterative dynamical methodology.

For specifically C7, i.e., chart summarization, we loosely consider both the determination of the problem statement pertaining to automatically interpreting chart images and the design of components C1–C6 of our workflow as the precondition phase. Here, we elaborate on the core and the analysis phases.

Core phase Here, we implemented the following:

- *Discover* Problem Characterization and Abstraction—this stage entails gathering *requirements* from experts and users. Thus, to understand the requirement of blind users' usability of automated chart interpretation with chart summary to improve accessibility to the visually impaired, we listened to experts in education for the visually impaired (VI), one visually impaired person, and volunteers who create educational resources for the VI, such as haptics, braille textbooks. We selected this group of experts and users in the context of interpreting charts given in high school mathematics textbooks. We gathered that the manually created summary currently includes visual characteristics of color, geometry, and details of chart title, axes, and ranges. The summary also optionally includes apparent characteristics of the data, and usually, the chart accompanies the data table, which can be read using screen readers. Since our goal is to improve accessibility to charts in print and mass media, where data table is usually unavailable to the user, we added this as an additional requirement.
- Design Encoding—To satisfy the requirements, we arrived at a sentence structure for specifically bar charts and their subtypes [6], using a flowchart (Fig. 5). This well-built sentence structure serves as the predefined template for the summary. It contained the *visual* summary of the appearance of the chart and the *data table* summary capturing the chart title, axes, and statistical descriptors. The descriptors include data distribution, range, mean, median, mode, standard deviation, and correlation values of attributes based on the subtype of bar charts. The summary also includes the variable-based trend patterns in the chart image in the case of ordinal attributes.
- Implement Prototype—We implemented the chart summarizer, i.e., C7 in the workflow (Fig. 1), as explained in section "The 7-Component Workflow". These *system-generated* summaries are programmatically generated for all the charts in our dataset.
- Deploy Release and Gather Feedback—We then released a set of sample chart images and its corresponding system-generated summaries in a user study to gather feedback. Since the purpose of the user study is to evaluate the completeness of the system-generated summary, visually abled participants are used in the feedback process.

User Study We conducted a user study to collect feedback on the initial system-generated summaries of chart images. This study is intended to refine the sentence structure in the predefined template for the chart summary. Our user study is similar to that conducted for identifying the intended message of a graphic [18], where the users evaluate our system-generated summary.

The core part of our user study has been designed as two different modules based on the objectives of the study. The first objective is to learn about the understanding and interpretation of bar chart images by the user. The second objective is to assess the completeness and effectiveness of the initial system-generated summary and help us further improve it. The second objective is partially similar to the user study for identifying the intended message of a simple bar chart [18], where participants are asked to rate specific propositions, i.e., pieces of information, as "essential," "possible," and "not important".

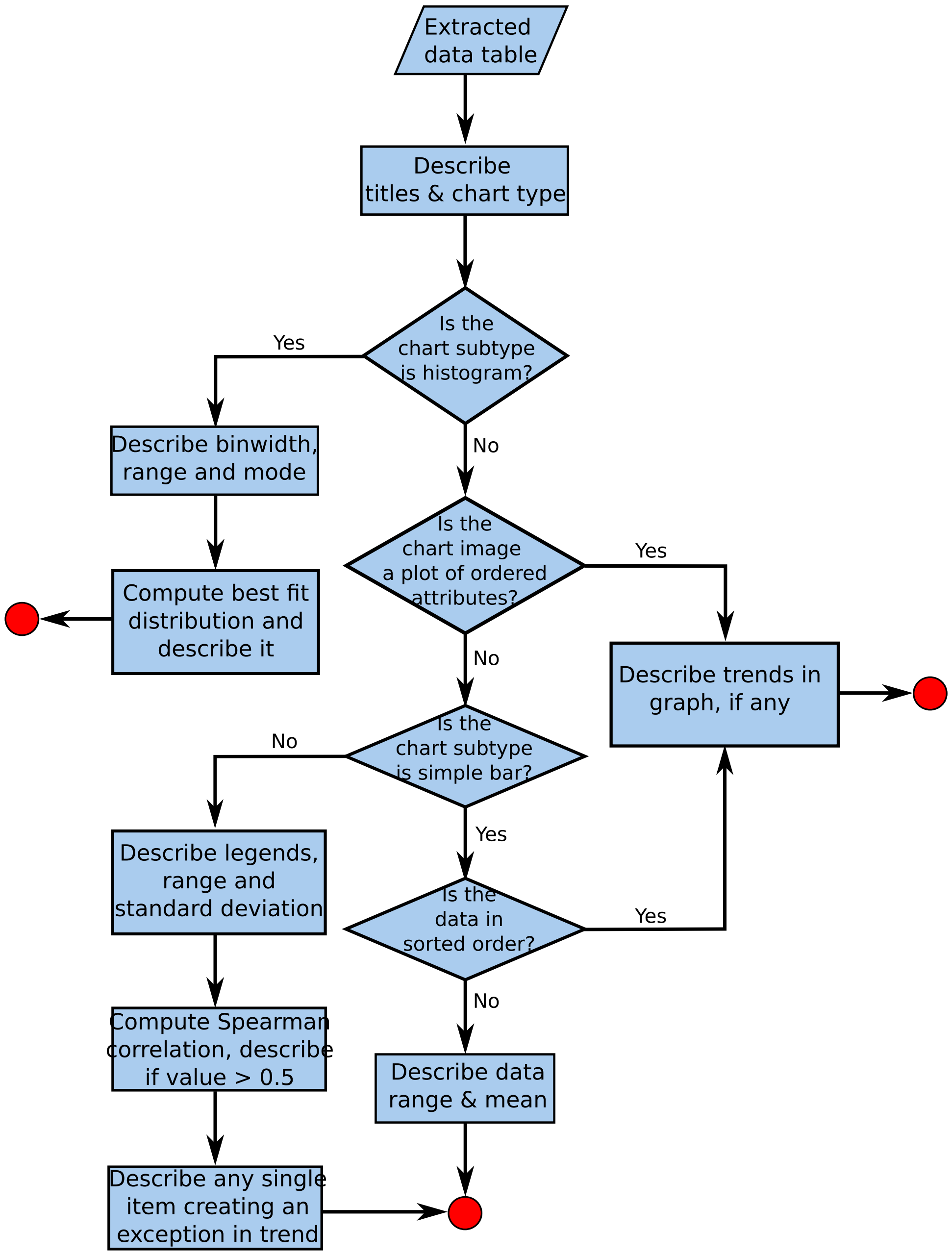
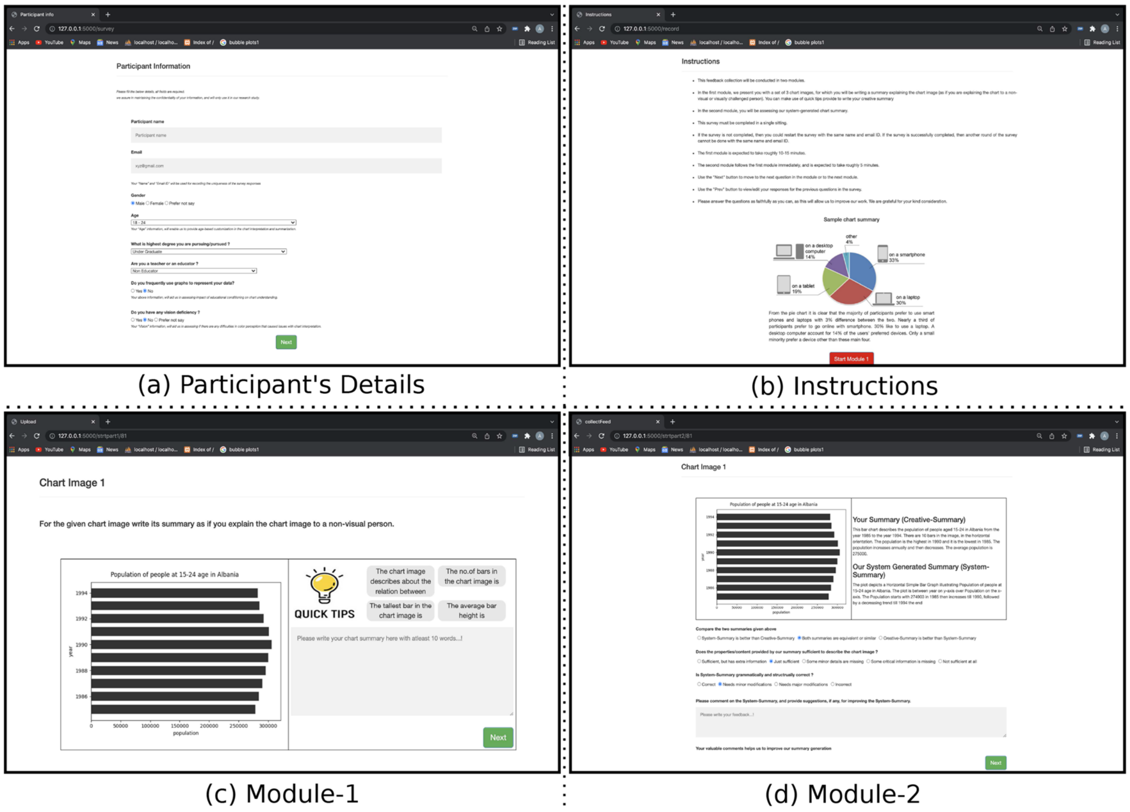

Extracted data table
Describe titles & chart type
Is the chart subtype is histogram?
Yes
No
Describe binwidth, range and mode
Compute best fit distribution and describe it
Is the chart image a plot of ordered attributes?
Yes
No
Describe trends in graph, if any
Is the chart subtype is simple bar?
No
Yes
Describe legends, range and standard deviation
Is the data in sorted order?
Yes
No
Compute Spearman correlation, describe if value > 0.5
Describe data range & mean
Describe any single item creating an exception in trend

Fig. 5 Our proposed flowchart of sentence structure formation in the chart summary (Source: [6]), that is implemented in the *design* stage of the core phase in the design study

We designed an online user study that allows us to collect diverse responses from various sections of the population. The online study is also required owing to the social distancing during the pandemic. The participants are invited to the study through email, social media, etc., and are informed about the purpose of the user study. The participants' responses are recorded using a web form on the study website. The web form has the following four parts in sequence, as shown in Fig. 6:

1. Participants' details—This information includes participant name, email, age, gender, educational degree, profession (if worked/working as an educator), vision deficiency, and self-assessment of familiarity with usage of charts. This information helps us analyze how factors like age, gender, educational conditioning, and vision deficiencies impact chart understanding and interpretation of participants.
2. Instructions—A web page of written detailed instructions is included in the study website, including the expected time commitment for the study. In addition to these instructions, further queries have been answered through telephonic calls, emails, social media correspondences, etc.
3. Module-1—The first module of the core part of the user study satisfies the first objective of the study. Here, each participant is shown a set of three bar chart images. These


(a) Participant's Details

Participant Information

Please fill the below details, all fields are required.
we assure in maintaining the confidentiality of your information, and will only use it in our research study.

Participant name: Participant name

Email: xyz@gmail.com

Your "Name" and "Email ID" will be used for recording the uniqueness of the survey responses

Gender: [✓ Male] Female Prefer not say

Age: 18 - 24

Your "Age" information, will enable us to provide age-based customization in the chart interpretation and summarization.

What is highest degree you are pursuing/pursued ? Under Graduate

Are you a teacher or an educator ? Non Educator

Do you frequently use graphs to represent your data? Yes [✓ No]

Your above information, will aid us in assessing impact of educational conditioning on chart understanding.

Do you have any vision deficiency ? Yes [✓ No] Prefer not say

Your "Vision" information, will aid us in assessing if there are any difficulties in color perception that caused issues with chart interpretation.

(b) Instructions

Instructions

- This feedback collection will be conducted in two modules.
- In the first module, we present you with a set of 3 chart images, for which you will be writing a summary explaining the chart image (as if you are explaining the chart to a non-visual or visually challenged person). You can make use of quick tips provide to write your creative summary
- In the second module, you will be assessing our system-generated chart summary.
- This survey must be completed in a single sitting.
- If the survey is not completed, then you could restart the survey with the same name and email ID. If the survey is successfully completed, then another round of the survey cannot be done with the same name and email ID.
- The first module is expected to take roughly 10-15 minutes.
- The second module follows the first module immediately, and is expected to take roughly 5 minutes.
- Use the "Next" button to move to the next question in the module or to the next module.
- Use the "Prev" button to view/edit your responses for the previous questions in the survey.
- Please answer the questions as faithfully as you can, as this will allow us to improve our work. We are grateful for your kind consideration.

Sample chart summary

From the pie chart it is clear that the majority of participants prefer to use smart phones and laptops with 3% difference between the two. Nearly a third of participants prefer to go online with smartphone. 30% like to use a laptop. A desktop computer account for 14% of the users' preferred devices. Only a small minority prefer a device other than these main four.

(c) Module-1

Chart Image 1

For the given chart image write its summary as if you explain the chart image to a non-visual person.

QUICK TIPS: The chart image describes about the relation between | The no.of bars in the chart image is | The tallest bar in the chart image is | The average bar height is

Please write your chart summary here with atleast 10 words...!

(d) Module-2

Chart Image 1

Your Summary (Creative-Summary)

This bar chart describes the population of people aged 15-24 in Albania from the year 1985 to the year 1994. There are 10 bars in the image, in the horizontal orientation. The population is the highest in 1990 and it is the lowest in 1985. The population increases annually and then decreases. The average population is 275000.

Our System Generated Summary (System-Summary)

The plot depicts a Horizontal Simple Bar Graph illustrating Population of people at 15-24 age in Albania. The plot is between year on y-axis over Population on the x-axis. The Population starts with 274903 in 1985 then increases till 1990, followed by a decreasing trend till 1994 the end

Compare the two summaries given above
System-Summary is better than Creative-Summary [✓ Both summaries are equivalent or similar] Creative-Summary is better than System-Summary

Does the properties/content provided by our summary sufficient to describe the chart image ?
Sufficient, but has extra information [✓ Just sufficient] Some minor details are missing Some critical information is missing Not sufficient at all

Is System-Summary grammatically and structrually correct ?
Correct [✓ Needs minor modifications] Needs major modifications Incorrect

Please comment on the System-Summary, and provide suggestions, if any, for improving the System-Summary.

Please write your feedback...!

Your valuable comments helps us to improve our summary generation

Fig. 6 A snapshot of user study displaying web page for **a** participant's details collection, **b** instructions of survey, **c** module-1 for creative summaries collection, and **d** module-2 for comparative assessment of summaries

images are randomly selected from 26 images from the test corpus. We ensure that the set presented to each participant includes a grouped bar chart, a stacked bar chart, and a simple bar/histogram. The participants see an image and complete the assigned task for the image before being shown the next image. The task for each image entails writing one's own *creative* chart summary for the image. The participant is advised to prepare the summary in a way he/she would narrate to a VI person.

4. Module-2—The second module of the core part of the user study satisfies the second objective of the study. Here, we present the user with the same set of chart images from the first module, their creative summaries by the participant prepared in Module-1, and the corresponding system-generated summaries from the *design* stage. These three entities are juxtaposed for the participant to compare the two summaries. The task entails comparing both the summaries and assessing using Likert-like scales. We use three components of the scores, i.e., compare-, sufficiency-, and grammar-scores, to rate our system-generated summaries qualitatively against their creative summaries. This task of comparison is done for one image at a time, as done in Module-1.

The three components of the total score used in our study are given below, with the range of rating in parentheses and the interpretation of each rating.

Compare-Score (1–3):

3. System-Summary is better than Creative-Summary
2. Both summaries are equivalent or similar
1. Creative-Summary is better than System-Summary.

Sufficiency-Score (1–5):

5. System-Summary is sufficient but has extra information
4. System-Summary is just sufficient
3. Some minor details are missing in System-Summary
2. Some critical information is missing in System-Summary
1. System-Summary is not sufficient at all.

Grammar-Score (1–4):

4. System-Summary is grammatically/structurally correct
3. System-Summary needs minor modifications
2. System-Summary needs major modifications

1. System-Summary is grammatically/structurally incorrect.

As a final step of the user study, participants are provided a text box in the web form to optionally provide short feedback with suggestions for improving the system-generated summary.

The 30 participants of the study, who were primarily undergraduate and graduate college students, were in the age group 18–45 and completed both Module-1 and 2. Each participant is given a set of three chart images with a grouped bar chart, a stacked bar chart, and a simple bar/histogram. These chart images are randomly selected from a generic test dataset of 26 images obtained from different sources like the internet and synthetically generated datasets that are simple and easily understood by most sections of the population. The outcome of the survey is that, in most of the cases, the participants were satisfied with the system-generated summary. They stated that the non-visual statistical description, such as distribution of histogram, standard and correlations values in the stacked and grouped bar, etc., improved their understanding of the data of the chart image. However, a few participants contradicted that the statistical description in the system-generated summary was extraneous. They also pointed out that the system-generated summaries had not included a few visually perceivable details, such as intra-class inferences, overall bar heights, and inter-class differences in heights in grouped and stacked bars. Some of the participants misinterpreted the histogram chart as a simple bar chart without considering the semantics of the bar being counts or frequencies and that of the chart being a visualization of the data distribution.

Apart from the suggestions on improving the content of the summary, there were suggestions on editorial corrections to the summaries, e.g., proper noun usage, capitalization of words, etc. These summaries also had instances of a few words being misspelled due to improper text recognition in C5. Some of the participants suggested that detailed data representation is desirable for charts with fewer bars, say < 8. There was another feedback to use *compound subjects* in sentences to shorten the summary, e.g., "A is correlated to B, B is correlated to C, and C is correlated to A" is to be rephrased as "A, B, and C are correlated to each other."

Quantifying the survey outcomes, the average grammar score is 3.33 out of 4.0, which states that minor grammatical modifications are required in initial summaries. The average compare-score is 2.33 out of 3.0, which states that our initial system-generated summaries cover similar details as the creative summaries by the participants. The participants reported that the system-generated summaries outperformed creative summaries, especially in cases of charts without any context, such as missing chart title or axes titles. The average sufficiency score is 3.97 out of 5.0, which states that system-generated summaries are sufficient. The outcomes of the user study lead to the analysis phase in the design study. The *reflect* stage of the analysis phase entails refining the summary using the feedback from the user study.

Analysis Phase: We have implemented the following steps given in the design study methodology (DSM) [7]:

- *Reflect*: Confirm and Refine—We confirmed our findings from the user study and then determined a change in the text organization, from the sentence flow format, for the predefined summary template. Our refined text organization is implemented using a new algorithm, as shown in Fig. 7. This algorithm creates summaries specific to each chart subtype, thus exploiting the semantics of the type. Here, we also use best practices of reusing components in programming, where the grouped and stacked bar charts are reduced to simple bar charts, and the summarizer for the simple bar chart is invoked. We refined the visual summary to include intra- and inter-class trends in grouped and stacked bar charts. For histograms, we re-annotate the variables to include terms such as "frequencies" and "bins" to convey the chart semantics. The new text organization now has three parts in sequence

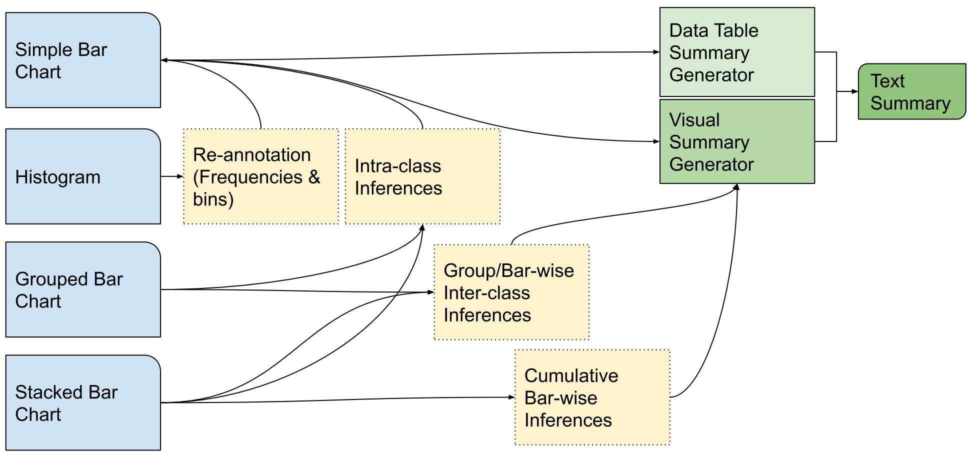

Simple Bar Chart
Histogram
Grouped Bar Chart
Stacked Bar Chart
Re-annotation (Frequencies & bins)
Intra-class Inferences
Group/Bar-wise Inter-class Inferences
Cumulative Bar-wise Inferences
Data Table Summary Generator
Visual Summary Generator
Text Summary

Fig. 7 Our proposed algorithm for generating the text summary in the BarChartAnalyzer, refined by the responses from the user study. This algorithm is implemented in the *reflect* stage of the analysis phase in the design study

[22]—the chart specifications, the visual summary, and the data table summary. The content from the visual and data table summaries are then de-duplicated to make the content crisp. We additionally used an open-source Python library, `GingerIt`, to correct the spelling and grammar mistakes based on the construction of the complete sentences. This gives us the final chart summary.
- *Write*: Design Study Paper—Our elaborate documentation in this section serves as the written report of the entire design study.

We conclude that our chart summarization provides a *master* summary. This summary can be further *pruned* based on user inputs, thus, customizing this component for *intended* purposes. The parts of the modified text organization can also be considered as the granularity of information, which can be used in an interactive framework for providing high-to-low level details of the graphic, e.g., Interactive SIGHT (Summarizing Information GrapHics Textually) [40].

Experiments and Results

This section discusses the overall implementation, and performance of the BarChartAnalyzer.

Implementation: The workflow is implemented in the `Pycharm 2018.3` tool on an Intel i5 processor equipped with 8 GB RAM in Mac OS 10.14.3. The manual intervention is required for image annotation in C2, and optionally for tuning hyperparameters for DBScan algorithm in C4. We manually annotate the chart image using the `LabelImg` tool separately installed on our machine, which takes $\sim$ 2 min. We then input the chart image along with the generated XML file to our BarChartAnalyzer. The execution time of the entire workflow of BarChartAnalyzer is $\sim$ 3 min for an image of 192 DPI (Dots Per Inch), of which the tensor voting computation consumes two-thirds of the entire running time. Our current implementation of the tensor voting computation is serial and has the scope of parallel implementation in future to reduce the execution time of BarChartAnalyzer.

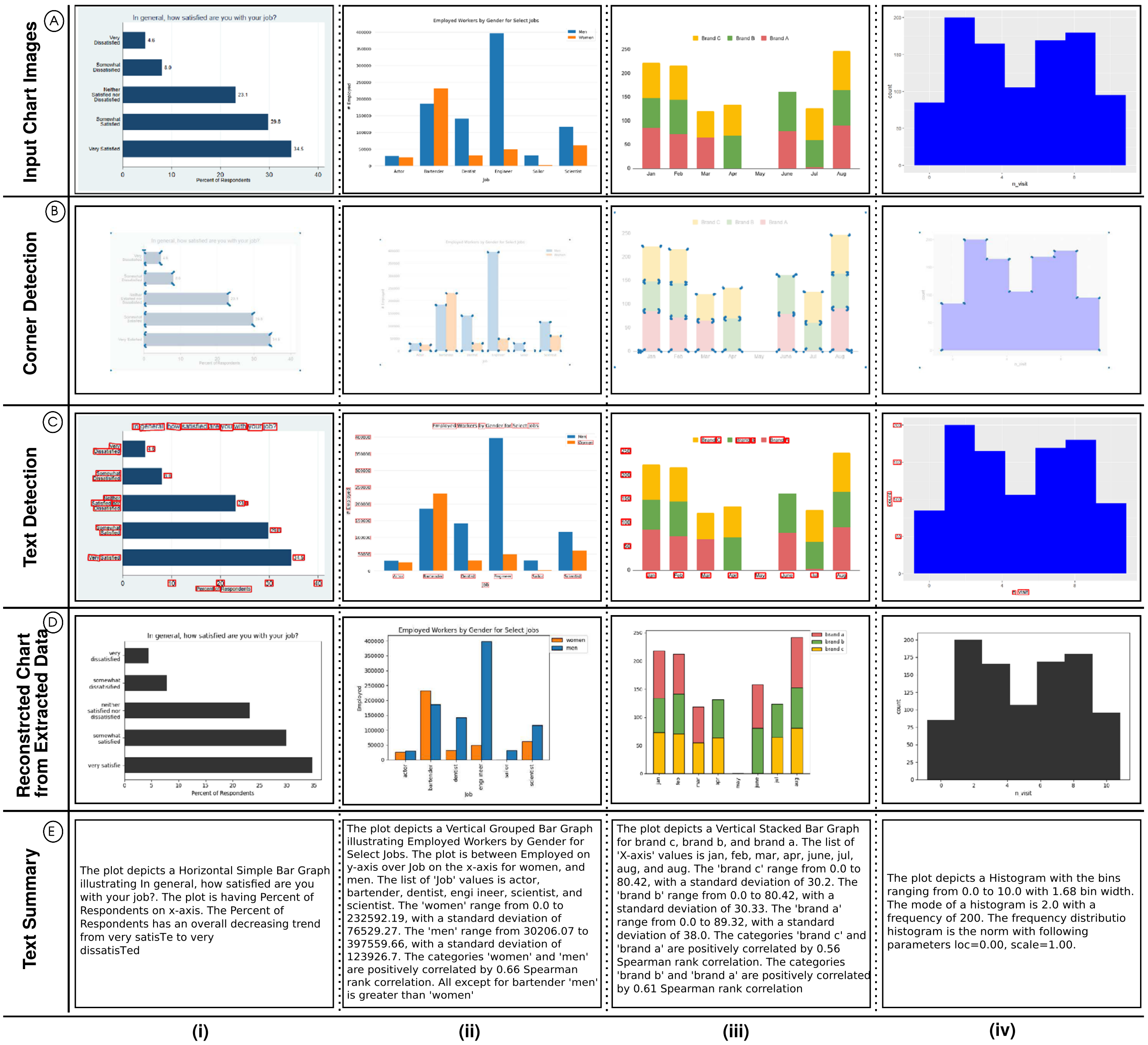
In C1 of the BarChartAnalyzer, we have trained the CNN model for classification using 1000 images belonging to eight types of charts, namely, the seven subtypes of bar charts and a complement set, “others”, consisting of chart images of line charts, scatter plots, and pie charts. The training set excludes images for charts with textured, hollow, or hand-drawn bar objects. The training accuracy for our classifier is currently at 85%. For testing, we have used a dataset of 50 chart images each from these eight types. For experiments, we generated a dataset that includes bar chart images of these eight types from two sources, namely images downloaded from the Internet and synthetically generated images. The latter is from bar charts generated using the Python plotting library, `matplotlib` from known data tables.

Experimental Results: We have tested our entire system on the 50 chart images for each of the eight bar chart subtypes, apart from using them for testing the chart type classifier. The results from the BarChartAnalyzer for a subset of our experiments are shown in Fig. 8. The images are first classified, and only those of bar charts and its subtypes pass through the BarChartAnalyzer. The source images are given in Fig. 8a. The tensor field analysis on extracted canvas detects the corner of the bars using critical points identified by the saliency value calculation. The results of pixels identified by corner detection are shown in Fig. 8b. The critical points are detected at the top and bottom corners of bars and at the bar segment junctions in the stacked bar chart. The histogram displays the distribution of such points at the junction where the transition between bins occurs. The visualization of critical points at corners also guides us in tuning the hyperparameters for DBSCAN, e.g., distance (*eps*), *minPts*.

The OCR-based text detection model [37] works with a 0.95 F1-score on ICDAR 2013 dataset. The model fails to detect certain text components during testing, as shown in Fig. 8c, ii. Our workflow addresses this limitation while performing data extraction based on pixel scaling and the intervals retrieved from the detected text/values. Our reconstructed charts in Fig. 8d can be visually compared with the original images in Fig. 8a.

Color is an important property of the images of the multi-class bar charts like grouped and stacked bar, as color is a visual encoding of the metadata of the classes. In such cases, color represents the identity of the classes the data item belongs to. However, in the case of simple bar charts and histograms, the use of color is cosmetic. Our algorithm preserves the source color value only in the case of it being a visual encoding, where we use the color value identified in the legend for the bars during reconstruction. In the cases where color is not used as a visual encoding for the chart, we use a default color value, i.e., black, during reconstruction. Thus, color is preserved in reconstruction for charts in Fig. 8ii, iii, but not in Fig. 8i, iv. However, even where color is preserved, the order of rendering the classes is not guaranteed to be preserved, as shown in Fig. 8ii, iii, as the ordering of the classes is not an important property in the multi-class bar charts.

Evaluation: The premise of our work is to extract data from images of charts that do not have accompanying data tables, i.e., the ground truth. Hence, to compare the extracted data with source information, we run our algorithm on images of charts generated using the plotting library, e.g., `matplotlib`, from known data tables. While our algorithm works well with such synthetically generated images owing to their high resolution and fidelity, they are



The plot depicts a Horizontal Simple Bar Graph illustrating In general, how satisfied are you with your job?. The plot is having Percent of Respondents on x-axis. The Percent of Respondents has an overall decreasing trend from very satisTe to very dissatisTed

The plot depicts a Vertical Grouped Bar Graph illustrating Employed Workers by Gender for Select Jobs. The plot is between Employed on y-axis over Job on the x-axis for women, and men. The list of 'Job' values is actor, bartender, dentist, engi ineer, scientist, and scientist. The 'women' range from 0.0 to 232592.19, with a standard deviation of 76529.27. The 'men' range from 30206.07 to 397559.66, with a standard deviation of 123926.7. The categories 'women' and 'men' are positively correlated by 0.66 Spearman rank correlation. All except for bartender 'men' is greater than 'women'

The plot depicts a Vertical Stacked Bar Graph for brand c, brand b, and brand a. The list of 'X-axis' values is jan, feb, mar, apr, june, jul, aug, and aug. The 'brand c' range from 0.0 to 80.42, with a standard deviation of 30.2. The 'brand b' range from 0.0 to 80.42, with a standard deviation of 30.33. The 'brand a' range from 0.0 to 89.32, with a standard deviation of 38.0. The categories 'brand c' and 'brand a' are positively correlated by 0.56 Spearman rank correlation. The categories 'brand b' and 'brand a' are positively correlated by 0.61 Spearman rank correlation

The plot depicts a Histogram with the bins ranging from 0.0 to 10.0 with 1.68 bin width. The mode of a histogram is 2.0 with a frequency of 200. The frequency distributio histogram is the norm with following parameters loc=0.00, scale=1.00.

Fig. 8 The key steps in our BarChartAnalyzer workflow of corner detection (C4), text detection (C5), data extraction (C6), chart reconstruction, and chart summary (C7) of the source input chart images. We observe that the ordering of the classes could be reversed in (ii) the grouped and (iii) stacked bar charts, even though the data are extracted accurately (Source: [6])

useful in computing exact numerical errors in the extracted data table.

The data extraction involves mapping the pixel location of the cluster center of degenerate points and text location extracted using OCR. Such a mapping causes the extracted values to have numerical precision errors predominantly. Hence, to compare the difference between the extracted values, we compute the normalized Mean Absolute Error (nMAE), and the Mean Absolute Percentage Error (MAPE) for the synthetic images (Fig. 9), which are bounded in [0,1]. MAPE is commonly reported in a percentage format. We observe that nMAE captures our performance better than MAPE, as it does not augment numerical precision errors as much as MAPE. MAPE is augmented in the case of missing extracted data in grouped bar charts (Fig. 9ii) and stacked bar charts (Fig. 9iii) owing to relatively short bars or bar segments. For N data items with source data value x_i and its corresponding extracted value $x_i^{(e)}$, for $i = 1, 2, \ldots, N$, we compute

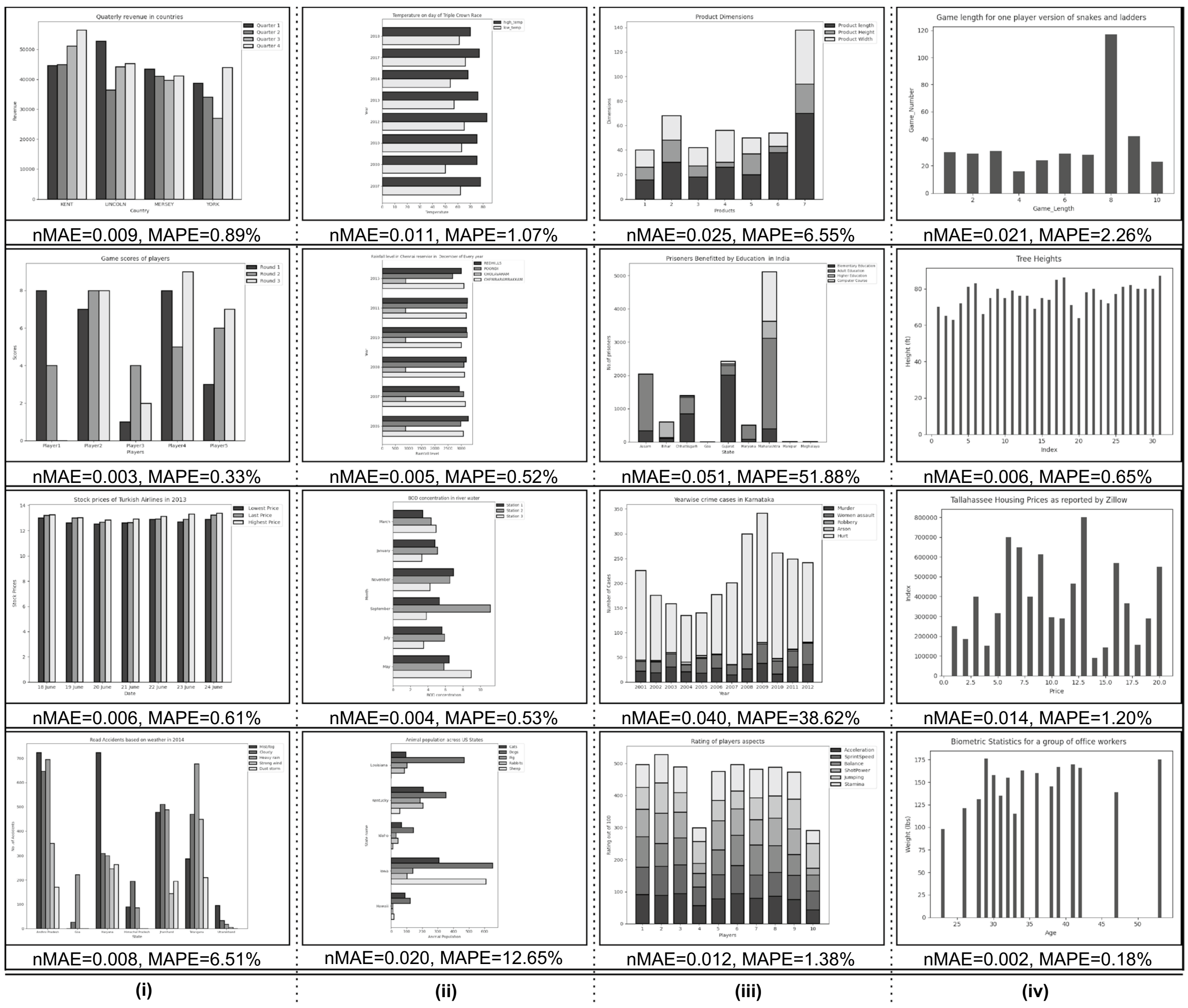


Fig. 9 Reconstruction of synthetically generated bar chart images with their error evaluation in normalized mean absolute error (nMAE) and mean absolute percentage error (MAPE) (Source: [6])

$$\mathrm{nMAE} = \frac{\sum_{i=1}^{N} |x_i - x_i^e|}{\sum_{i=1}^{N} x_i}. \tag{5}$$

In our representative examples in Fig. 9, we observe relatively low nMAE values. Histograms are not included in this analysis as the source, and extracted data in its case are a frequency table, different from a data table in the case of other subtypes of bar charts.

Table 2 gives an overall accuracy of the data extraction component of our BarChartAnalyzer, comparing them to the results reported by the state-of-the-art data extraction methods. Our work differentiates the performance of the bar chart subtypes, unlike the state-of-the-art methods. Owing to the low count of images used for computing the scores, a few *failures* in data extraction tend to bring the scores down. Hence, increasing the test corpus size for validation would give us more realistic comparative results. Despite the caveat of the test corpus size, our proposed workflow performs as comparable as the state-of-the-art data extraction methods. BarChartAnalyzer achieves near-perfect accuracy for high-resolution bar chart images, created with standard or minimal formatting available commonly across plotting libraries. The morphological methods for image preprocessing in C3 in BarChartAnalyzer improve data extraction accuracy from low-fidelity images. The aggregated accuracy for PlotQA [1] for CQA is 22%, and STL-CQA [14] achieves near-perfect accuracy, but with synthetic datasets. However, comparing our work with the CQA algorithms is not fair,

Table 2 Comparison of accuracy of data extraction of BarChartAnalyzer with the state-of-the-art

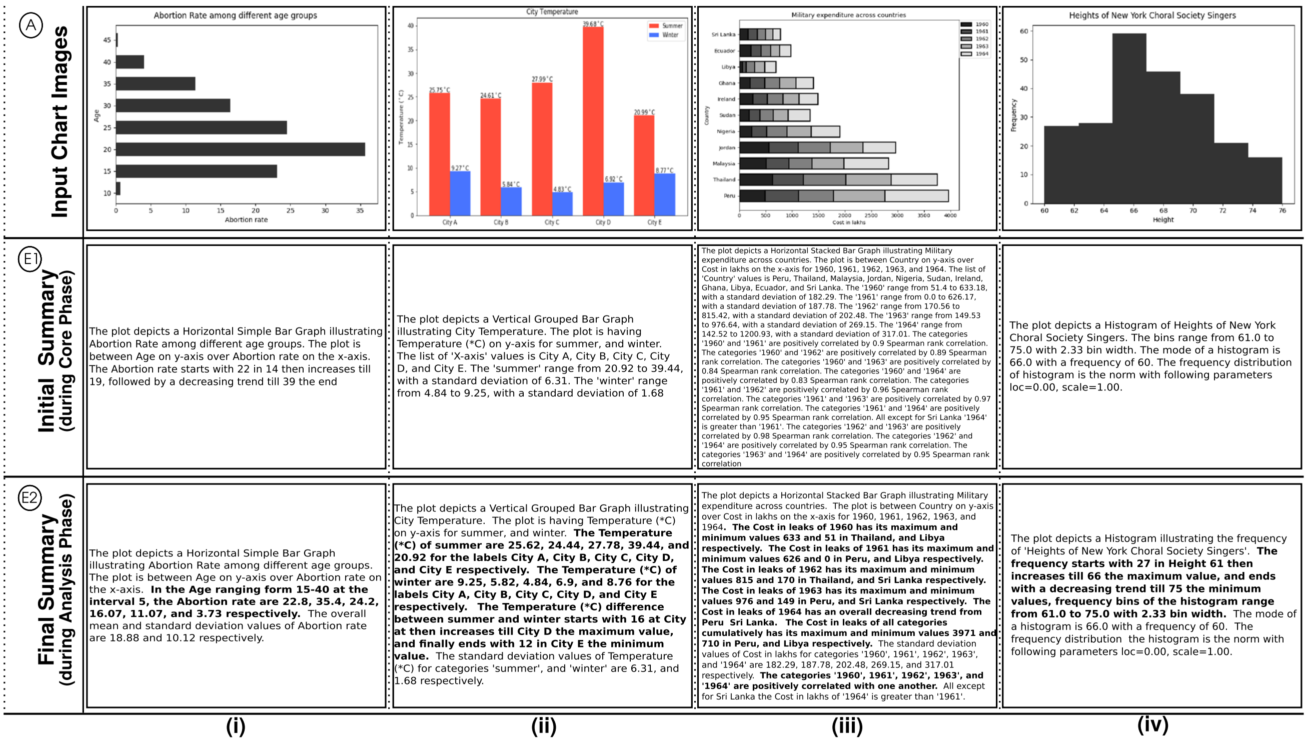
Method	Chart type	Precision (%)	Recall (%)	F1-score (%)
MECDG [3]	Bar (all)	**91.2**	**94.6**	**92.9**
ReVision [8]	Bar (all)	78.3	84.6	81.3
ChartSense [13]	Bar (all)	90.7	92.1	91.3
Choi et al. [4]	Bar (all)—figureQA	**93.5**	**94.0**	**93.7**
	Bar (all)—web-collected	**92.9**	64.5	76.1
Ours	Grouped bar	**100**	**99.2**	**99.6**
	Horizontal grouped bar	**100**	**98**	**98.9**
	Stacked bar	**98.8**	84.1	90
	Horizontal stacked bar	72.7	67.9	70
	(Simple) bar	75	67	70.5
	Horizontal bar	**100**	**100**	**100**
	$\boxed{\text{Bar (all)}}$	$\boxed{91}$	$\boxed{86}$	$\boxed{\mathbf{91.7}}$

The boldface highlights indicate values greater than the average results from our method in bold

as the goals are different, even though there are overlapping outcomes.

Figure 10 shows the sample results of our workflow for text summary generation for different bar chart types. We observe that the summaries are qualitatively complete after the design study. The summaries for grouped and stacked bar charts are longer and contain more details, serving well as the master summary.

Limitations: In a limited number of cases, our system suffers from errors in detection, specifically when DBSCAN clustering does not distinguish small/insignificant height differences between bars/bins [5]. We have identified two such cases. The first case is of false negatives when bars are close to the baseline, which is the x-axis and y-axis for column and bar orientations, respectively (Fig. 11d, ii). The second case is when heights of adjacent bins in a histogram have relatively small height



The plot depicts a Horizontal Simple Bar Graph illustrating Abortion Rate among different age groups. The plot is between Age on y-axis over Abortion rate on the x-axis. The Abortion rate starts with 22 in 14 then increases till 19, followed by a decreasing trend till 39 the end

The plot depicts a Vertical Grouped Bar Graph illustrating City Temperature. The plot is having Temperature (*C) on y-axis for summer, and winter. The list of 'X-axis' values is City A, City B, City C, City D, and City E. The 'summer' range from 20.92 to 39.44, with a standard deviation of 6.31. The 'winter' range from 4.84 to 9.25, with a standard deviation of 1.68

The plot depicts a Horizontal Stacked Bar Graph illustrating Military expenditure across countries. The plot is between Country on y-axis over Cost in lakhs on the x-axis for 1960, 1961, 1962, 1963, and 1964. The list of 'Country' values is Peru, Thailand, Malaysia, Jordan, Nigeria, Sudan, Ireland, Ghana, Libya, Ecuador, and Sri Lanka. The '1960' range from 51.4 to 633.18, with a standard deviation of 182.29. The '1961' range from 0.0 to 626.17, with a standard deviation of 187.78. The '1962' range from 170.56 to 815.42, with a standard deviation of 202.48. The '1963' range from 149.53 to 976.64, with a standard deviation of 269.15. The '1964' range from 142.52 to 1200.93, with a standard deviation of 317.01. The categories '1960' and '1961' are positively correlated by 0.9 Spearman rank correlation. The categories '1960' and '1962' are positively correlated by 0.89 Spearman rank correlation. The categories '1960' and '1963' are positively correlated by 0.84 Spearman rank correlation. The categories '1960' and '1964' are positively correlated by 0.83 Spearman rank correlation. The categories '1961' and '1962' are positively correlated by 0.96 Spearman rank correlation. The categories '1961' and '1963' are positively correlated by 0.97 Spearman rank correlation. The categories '1961' and '1964' are positively correlated by 0.95 Spearman rank correlation. All except for Sri Lanka '1964' is greater than '1961'. The categories '1962' and '1963' are positively correlated by 0.98 Spearman rank correlation. The categories '1962' and '1964' are positively correlated by 0.95 Spearman rank correlation. The categories '1963' and '1964' are positively correlated by 0.95 Spearman rank correlation

The plot depicts a Histogram of Heights of New York Choral Society Singers. The bins range from 61.0 to 75.0 with 2.33 bin width. The mode of a histogram is 66.0 with a frequency of 60. The frequency distribution of histogram is the norm with following parameters loc=0.00, scale=1.00.

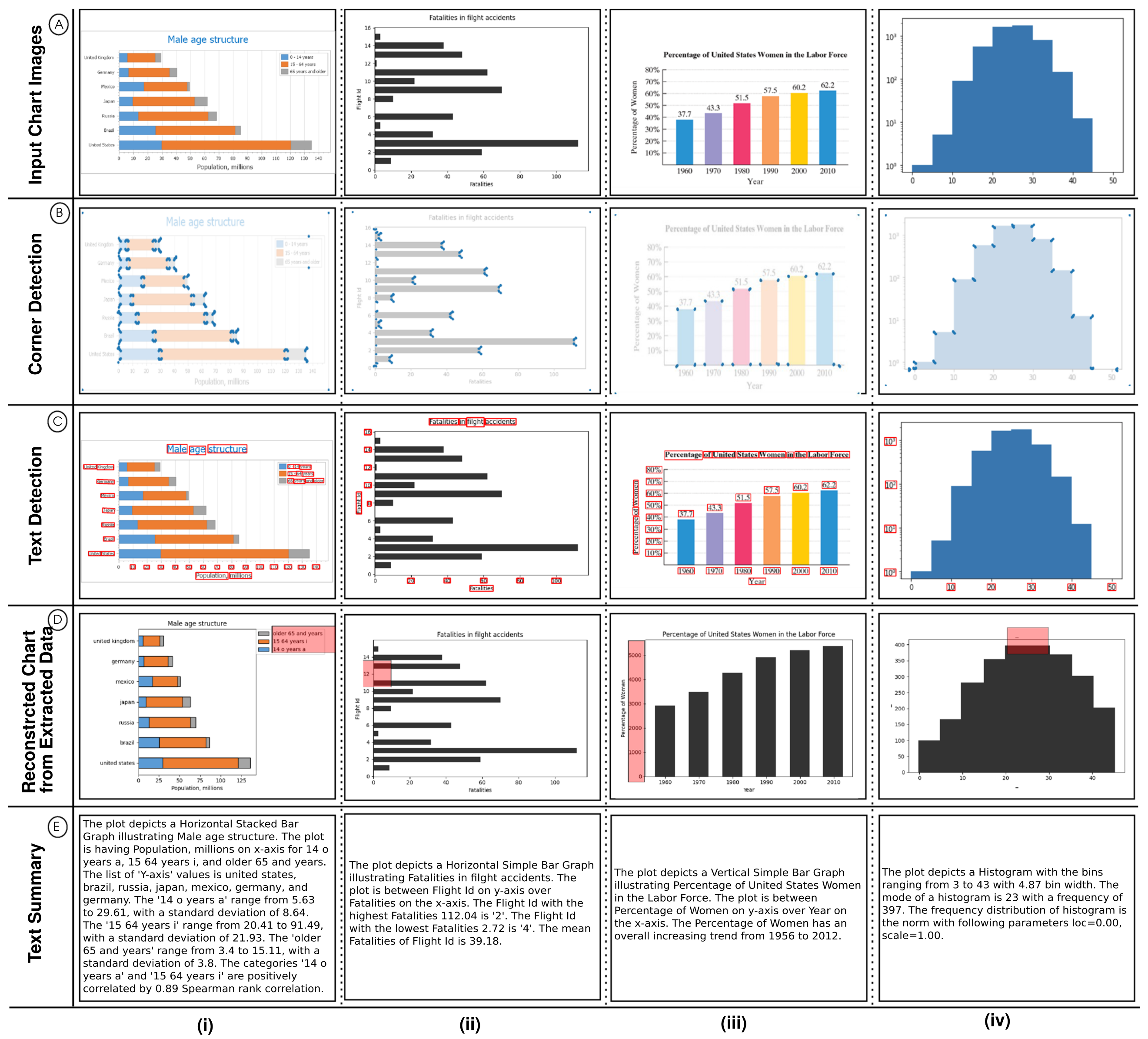
The plot depicts a Horizontal Simple Bar Graph illustrating Abortion Rate among different age groups. The plot is between Age on y-axis over Abortion rate on the x-axis. **In the Age ranging form 15-40 at the interval 5, the Abortion rate are 22.8, 35.4, 24.2, 16.07, 11.07, and 3.73 respectively.** The overall mean and standard deviation values of Abortion rate are 18.88 and 10.12 respectively.

The plot depicts a Vertical Grouped Bar Graph illustrating City Temperature. The plot is having Temperature (*C) on y-axis for summer, and winter. **The Temperature (*C) of summer are 25.62, 24.44, 27.78, 39.44, and 20.92 for the labels City A, City B, City C, City D, and City E respectively. The Temperature (*C) of winter are 9.25, 5.82, 4.84, 6.9, and 8.76 for the labels City A, City B, City C, City D, and City E respectively. The Temperature (*C) difference between summer and winter starts with 16 at City at then increases till City D the maximum value, and finally ends with 12 in City E the minimum value.** The standard deviation values of Temperature (*C) for categories 'summer', and 'winter' are 6.31, and 1.68 respectively.

The plot depicts a Horizontal Stacked Bar Graph illustrating Military expenditure across countries. The plot is between Country on y-axis over Cost in lakhs on the x-axis for 1960, 1961, 1962, 1963, and 1964**. The Cost in leaks of 1960 has its maximum and minimum values 633 and 51 in Thailand, and Libya respectively. The Cost in leaks of 1961 has its maximum and minimum values 626 and 0 in Peru, and Libya respectively. The Cost in leaks of 1962 has its maximum and minimum values 815 and 170 in Thailand, and Sri Lanka respectively. The Cost in leaks of 1963 has its maximum and minimum values 976 and 149 in Peru, and Sri Lanka respectively. The Cost in leaks of 1964 has an overall decreasing trend from Peru Sri Lanka. The Cost in leaks of all categories cumulatively has its maximum and minimum values 3971 and 710 in Peru, and Libya respectively.** The standard deviation values of Cost in lakhs for categories '1960', 1961', 1962', 1963', and '1964' are 182.29, 187.78, 202.48, 269.15, and 317.01 respectively. **The categories '1960', 1961', 1962', 1963', and '1964' are positively correlated with one another.** All except for Sri Lanka the Cost in lakhs of '1964' is greater than '1961'.

The plot depicts a Histogram illustrating the frequency of 'Heights of New York Choral Society Singers'. **The frequency starts with 27 in Height 61 then increases till 66 the maximum value, and ends with a decreasing trend till 75 the minimum values, frequency bins of the histogram range from 61.0 to 75.0 with 2.33 bin width.** The mode of a histogram is 66.0 with a frequency of 60. The frequency distribution the histogram is the norm with following parameters loc=0.00, scale=1.00.

Fig. 10 Comparison of the summaries generated during the core and analysis phases of our design study for chart summarization. The boldface formatted excerpts in row E2 are added during the *reflect* stage of the analysis phase in the design study



The plot depicts a Horizontal Stacked Bar Graph illustrating Male age structure. The plot is having Population, millions on x-axis for 14 o years a, 15 64 years i, and older 65 and years. The list of 'Y-axis' values is united states, brazil, russia, japan, mexico, germany, and germany. The '14 o years a' range from 5.63 to 29.61, with a standard deviation of 8.64. The '15 64 years i' range from 20.41 to 91.49, with a standard deviation of 21.93. The 'older 65 and years' range from 3.4 to 15.11, with a standard deviation of 3.8. The categories '14 o years a' and '15 64 years i' are positively correlated by 0.89 Spearman rank correlation.

The plot depicts a Horizontal Simple Bar Graph illustrating Fatalities in flight accidents. The plot is between Flight Id on y-axis over Fatalities on the x-axis. The Flight Id with the highest Fatalities 112.04 is '2'. The Flight Id with the lowest Fatalities 2.72 is '4'. The mean Fatalities of Flight Id is 39.18.

The plot depicts a Vertical Simple Bar Graph illustrating Percentage of United States Women in the Labor Force. The plot is between Percentage of Women on y-axis over Year on the x-axis. The Percentage of Women has an overall increasing trend from 1956 to 2012.

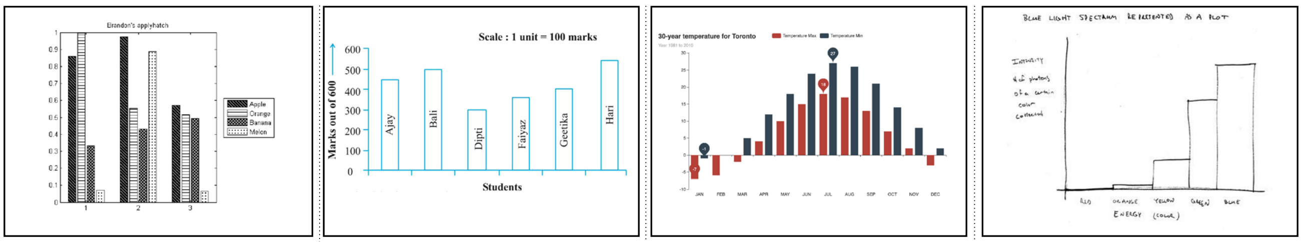
The plot depicts a Histogram with the bins ranging from 3 to 43 with 4.87 bin width. The mode of a histogram is 23 with a frequency of 397. The frequency distribution of histogram is the norm with following parameters loc=0.00, scale=1.00.

Fig. 11 Examples of bar chart images that give erroneous results in BarChartAnalyzer (Source: [6]). The errors in the chart reconstruction are indicated using red translucent boxes in row D

differences, and the extracted data do not capture the differences (Fig. 11d, iv). This error is also manifested as missing values in grouped and stacked bar charts when the bars or bar segments are relatively short (Fig. 9ii, iii).

The text recognition model [38] identifies text with an F1-score of 0.93 on ICDAR 2013 dataset. This recognition model misidentifies and confuses the alphabet ’O’ or ’o’, irrespective of the case, as the numeral ’0’ and vice versa in chart images. Also, the model has gaps in handling special characters, such as \$,%,£, sign(−), and cannot handle superscript symbols, e.g., degrees, and exponents (Fig. 11c, iii). These shortcomings affect the accuracy of the extracted data scale (Fig. 11c, i). The inaccurate results in text recognition also manifest as errors in the textual summary of the source image. Some of these errors in text detection are shown in the reconstructed chart in Figs. 11d, i and d, iii. The accuracy of our chart summarization is limited by that of text recognition, as expected [16].

One of the critical drawbacks of our BarChartAnalyzer is in the false positives for corner detection in relatively low-fidelity images, owing to aliasing and subsequent pixelation. Our method, in combination with more recent deep-learning methods for corner detection [3], may alleviate this issue. Also, our classification model cannot handle variants of bar charts with textures in the bars or hollow bars. Our workflow fails for chart canvas extraction for such exceptional test cases,



(a) Textured grouped bar (b) Hollow bars with text inside (c) Bar chart with graphics (d) Hand drawn bar chart

Fig. 12 Bar charts generated in different design spaces, which are known to not work with our chart analysis workflow, BarChartAnalyzer (Source: [6])

e.g., images shown in Fig. 12. Even though not widely practiced, bars can be re-annotated, e.g., by including text or bar value written inside each bar (Fig. 12b), which cannot be analyzed using BarChartAnalyzer. BarChartAnalyzer also fails for another test case where the data extraction process cannot interpret bar charts with divergent axes, e.g., presence of negative bars (Fig. 12c). Our text recognition model fails to identify text written in the hand-drawn chart shown in Fig. 12d. With respect to corner detection, BarChartAnalyzer fails in a few cases owing to the errors in human-guided annotation, critical point detection in tensor fields, and hyperparameter setting for DBSCAN for clustering corner points.

Improving chart summarization further requires an iterative process of feedback gathering through a user study in the *deploy* stage and the refinement of the summary in the *reflect* stage in our proposed design study. Hence, a careful design of subsequent user studies is in the scope of our future work. Several of the state-of-the-art methods use bilingual evaluation understudy (BLEU) [41] for validation of the chart summaries [3, 25]. In the absence of a ground truth or gold summary, we have not used the BLEU evaluation in our work. Even though we applied BLEU with respect to the creative summaries of users, BLEU underperformed owing to the variety and language styles in the creative summaries. An alternative that can be pursued in the future is to manually generate gold summaries for all chart images and to compute scores of the system-generated summaries with respect to the gold summaries based on content selection, relation generation, and content ordering metrics [42].

It is an interesting exercise to provide a confidence score upstream in our system on the success of automating data extraction for any input chart image. This requires running our system for a larger set of images from the different subtypes, and performing an in-depth analysis on the causality of the failures. This analysis is in the scope of future work of improving the usability of the BarChartAnalyzer system.

Conclusions

As a next step, a subtype-based analysis can be extended to other chart types, such as scatter plots. Our workflow requires user interaction for tasks such as image annotation for canvas extraction and setting hyperparameters of DBSCAN. In the scope of future work, the automation in our workflow can be increasing using more CNN models for annotation. We currently use tensor field computation on the chart images, which can be made more robust to separate chart objects from the source image. As VGGNet has been widely used for object detection tasks, our goal is to improve our classifier to automate the canvas extraction step, thus making our end-to-end workflow completely automated. Super-resolution algorithms may be explored as an additional component in our algorithm to improve the accuracy of both OCR and object detection, especially for severely aliased images.

In summary, we propose a workflow BarChartAnalyzer using standard image processing techniques and deep-learning models to perform the critical task of chart image digitization and summarization for bar charts. BarChartAnalyzer is novel in handling seven different bar chart subtypes. Our contributions include the mapping between pixel space data and the data space using the text detection model. We propose a novel application of the design study methodology for chart summarization. The design study provides the predefined template with information from the chart type details, perceivable visual features in the chart, and the data extracted from the chart image. The summarization achieved from our system has the potential of being used in a language processing module, such as `gtts` in Python, to generate an audio summary of the given chart image for the visually impaired audience. Our text summary can be pruned to provide crisper summaries as per user inputs. As discussed, our workflow has limitations of the dependency of workflow on the image fidelity, object size, training dataset, a variety of chart images, etc. Overall, our work shows promising results in automatically interpreting the chart images for bar charts and its subtypes, mostly human-out-of-the-loop approaches.

Acknowledgements This work has been funded by the Machine Intelligence and Robotics Center (MINRO) grant to the International Institute of Information Technology Bangalore (IIITB) by the Government of Karnataka. The authors are grateful for the discussion with T. K. Srikanth, IIITB; Sindhu Mathai of Azim Premji University; Vidhya Y. and Supriya Dey of Vision Empower; Neha Trivedi, XRCVC; Vani, Pushpaja, Kalyani, and Anjana of Braille Resource Center, Matruchayya, that has shaped this work. The authors are thankful for the helpful comments from anonymous reviewers.

Data Availability The curated training dataset of bar chart images contributed by this work is at https://github.com/GVCL/GVCL.github.io/tree/master/static/data/Test.

Declarations

Conflict of Interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Methani N, Ganguly P, Khapra MM, Kumar P. PlotQA: reasoning over scientific plots. In: The IEEE Winter Conference on Applications of Computer Vision. 2020;1516–1525.
2. Burns R, Carberry S, Elzer S. Modeling relative task effort for grouped bar charts. In: Proceedings of the Annual Meeting of the Cognitive Science Society. 2009;31:2292–2297.
3. Chen L, Zhao K. An approach for chart description generation in cyber-physical-social system. Symmetry. 2021;13(9):1552.
4. Choi J, Jung S, Park DG, Choo J, Elmqvist N. Visualizing for the Non-Visual: Enabling the Visually Impaired to Use Visualization. In: Computer Graphics Forum. vol. 38. Wiley Online Library; 2019:249–260.
5. Sreevalsan-Nair J, Dadhich K, Daggubati SC. Tensor fields for data extraction from chart images: bar charts and scatter plots. In: Hotz I, Bin Masood T, Sadlo F, Tierny J, editors. Topological methods in data analysis and visualization VI. Cham: Springer; 2021. p. 219–241. arXiv:2010.02319.
6. Dadhich K, Daggubati SC, Sreevalsan-Nair J. BarChartAnalyzer: digitizing images of bar charts. In: Proceedings of 1st International Conference on Image Processing and Vision Engineering (IMPROVE). INSTICC: SciTePress; 2021. p. 17–28.
7. Sedlmair M, Meyer M, Munzner T. Design study methodology: reflections from the trenches and the stacks. IEEE Trans Vis Comput Gr. 2012;18(12):2431–40.
8. Savva M, Kong N, Chhajta A, Fei-Fei L, Agrawala M, Heer J. ReVision: Automated classification, analysis and redesign of chart images. In: Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology. UIST '11. New York: Association for Computing Machinery; 2011. p. 393–402. https://doi.org/10.1145/2047196.2047247.
9. Rohatgi A. WebPlotDigitizer.
10. Battle L, Duan P, Miranda Z, Mukusheva D, Chang R, Stonebraker M. Beagle: automated extraction and interpretation of visualizations from the web. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems. CHI '18. New York: Association for Computing Machinery; 2018. p. 1–8. https://doi.org/10.1145/3173574.3174168.
11. Poco J, Heer J. Reverse-engineering visualizations: recovering visual encodings from chart images. In: Computer Graphics Forum. vol. 36. Wiley Online Library; 2017:353–363.
12. Siegel N, Horvitz Z, Levin R, Divvala S, Farhadi A. FigureSeer: parsing result-figures in research papers. In: Leibe B, Matas J, Sebe N, Welling M, editors. Comput. Vis. ECCV 2016. Cham: Springer International Publishing; 2016. p. 664–80.
13. Jung D, Kim W, Song H, Hwang Ji, Lee B, Kim B, et al. ChartSense: Interactive data extraction from chart images. In: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems. CHI '17. New York: Association for Computing Machinery; 2017. p. 6706–6717. https://doi.org/10.1145/3025453.3025957.
14. Singh H, Shekhar S. STL-CQA: structure-based transformers with localization and encoding for chart question answering. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020:3275–3284.
15. Al-Zaidy RA, Giles CL. Automatic extraction of data from bar charts. In: Proceedings of the 8th International Conference on Knowledge Capture. K-CAP 2015. New York: Association for Computing Machinery; 2015. p. 1–4. https://doi.org/10.1145/2815833.2816956.
16. Davila K, Setlur S, Doermann D, Bhargava UK, Govindaraju V. Chart mining: a survey of methods for automated chart analysis. IEEE Trans Pattern Anal Machine Intell. 2020.
17. Zhao J, Fan M, Feng M. Chartseer: Interactive steering exploratory visual analysis with machine intelligence. IEEE Trans Vis Comput Gr. 2020.
18. Demir S, Carberry S, McCoy KF. Summarizing information graphics textually. Comput Linguist. 2012;38(3):527–74.
19. Ferres L, Verkhogliad P, Lindgaard G, Boucher L, Chretien A, Lachance M. Improving accessibility to statistical graphs: the IGraph-lite system. In: Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility. Assets '07. New York: Association for Computing Machinery; 2007. p. 67–74. https://doi.org/10.1145/1296843.1296857.
20. Al-Zaidy R, Choudhury S, Giles C. Automatic summary generation for scientific data charts. In: WS-16-01. vol. WS-16-01 - WS-16-15. United States: AI Access Foundation; 2016. p. 658–663.
21. Demir S, Carberry S, McCoy KF. Generating Textual Summaries of Bar Charts. In: Proceedings of the Fifth International Natural Language Generation Conference. INLG '08. USA: Association for Computational Linguistics; 2008. p. 7–15.
22. Moraes P, Sina G, McCoy K, Carberry S. Evaluating the accessibility of line graphs through textual summaries for visually impaired users. In: Proceedings of the 16th International ACM SIGACCESS Conference on Computers & Accessibility. ACM; 2014. p. 83–90.
23. Xu S, Bryan C, Li JK, Zhao J, Ma KL. Chart Constellations: Effective Chart Summarization for Collaborative and Multi-User Analyses. In: Computer Graphics Forum. vol. 37. Wiley Online Library; 2018. p. 75–86.
24. Liu C, Xie L, Han Y, Wei D, Yuan X. Autocaption: An approach to generate natural language description from visualization automatically. In: Proceedings of 2020 IEEE Pacific Visualization Symposium (PacificVis). IEEE; 2020. p. 191–195.
25. Obeid J, Hoque E. Chart-to-Text: generating natural language descriptions for charts by adapting the transformer model. In: Proceedings of the 13th International Conference on Natural Language Generation. Dublin, Ireland: Association for Computational Linguistics; 2020. p. 138–147. https://aclanthology.org/2020.inlg-1.20.
26. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L, ImageNet: a large-scale hierarchical image database. In: 2009 IEEE conference

on computer vision and pattern recognition. IEEE; 2009. p. 248–255.

27. Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In: Bengio Y, LeCun Y, editors. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings; 2015. arXiv:1409.1556.
28. Chen T. Going deeper with convolutional neural network for intelligent transportation. Ph. D. Dissertation, Dept. Elect. Comput. Engg., Worcester Polytechnic Institute, 2015.
29. Fritsch J, Kuehnl T, Geiger A. A new performance measure and evaluation benchmark for road detection algorithms. In: 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013). IEEE; 2013. p. 1693–1700.
30. Tzutalin.: LabelImg. GitHub. https://github.com/tzutalin/labelImg. Accessed 3 Mar, 2022.
31. Medioni G, Tang CK, Lee MS. Tensor voting: theory and applications. In: Proceedings of RFIA, Paris, France; 2000.
32. Sreevalsan-Nair J, Kumari B. In: Schulz T, Özarslan E, Hotz I, editors. Local geometric descriptors for multi-scale probabilistic point classification of airborne LiDAR point clouds. Mathematics and Visualization. Cham: Springer; 2017. p. 175–200.
33. Wu TP, Yeung SK, Jia J, Tang CK, Medioni G. A closed-form solution to tensor voting: theory and applications. arXiv:1601.04888; 2016. p. 1–17.
34. Moreno R, Pizarro L, Burgeth B, Weickert J, Garcia MA, Puig D. Adaptation of tensor voting to image structure estimation. In: Laidlaw DH, Vilanova A, editors. New developments in the visualization and processing of tensor fields. Berlin: Springer; 2012. p. 29–50.
35. Wang S, Hou T, Li S, Su Z, Qin H. Anisotropic elliptic PDEs for feature classification. Visualization and computer graphics. IEEE Trans. 2013;19(10):1606–18. https://doi.org/10.1109/TVCG.2013.60.
36. Ester M, Kriegel HP, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. KDD'96. Palo Alto: AAAI Press; 1996. p. 226–231.
37. Baek Y, Lee B, Han D, Yun S, Lee H. Character region awareness for text detection. In: 2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR); 2019. p. 9357–9366.
38. Baek J, Kim G, Lee J, Park S, Han D, Yun S, et al. What is wrong with scene text recognition model comparisons? Dataset Model Anal. 2019;4714–4722.
39. Smith R. An Overview of the Tesseract OCR Engine. In: Ninth international conference on document analysis and recognition (ICDAR 2007). vol. 2. IEEE; 2007. p. 629–633.
40. Demir S, Oliver D, Schwartz E, Elzer S, Carberry S, McCoy KF. Interactive sight into information graphics. In: Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A); 2010. p. 1–10.
41. Papineni K, Roukos S, angard T, Zhu WJ. BLEU: A method for automatic evaluation of machine translation. In: Proceedings of the 40th annual meeting of the Association for Computational Linguistics; 2002. p. 311–318.
42. Wiseman S, Shieber SM, Rush AM. Challenges in data-to-document generation. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Copenhagen, Denmark: Association for Computational Linguistics; 2017. p. 2253–2263. https://aclanthology.org/D17-1239.