Titre: Title:	From Graphs to Words: A Computer-Assisted Framework for the Production of Accessible Text Descriptions
Auteur: Author:	Qiang Xu
Date:	2024
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Xu, Q. (2024). From Graphs to Words: A Computer-Assisted Framework for the Production of Accessible Text Descriptions [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/59637/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/59637/
Directeurs de recherche: Advisors:	Thomas Hurtut
Programme: Program:	Génie informatique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

From Graphs to Words: A Computer-Assisted Framework for the Production of Accessible Text Descriptions

QIANG XU
Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie informatique

Octobre 2024

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

From Graphs to Words: A Computer-Assisted Framework for the Production of Accessible Text Descriptions

présenté par **Qiang XU**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Jinghui CHENG, président
Thomas HURTUT, membre et directeur de recherche
Philippe DOYON-POULIN, membre

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to my supervisor for his acceptance, guidance, and understanding. I am also thankful to the representatives and participants from the granting agency and companies for their assistance throughout various stages of the project, as well as for their role in organizing and participating in the user study. On a personal note, I owe my deepest appreciation to my cat and my husband, whose unconditional support has been my greatest source of strength.

RÉSUMÉ

Dans le paysage numérique, l'omniprésence des visualisations de données dans les médias souligne la nécessité de l'accessibilité afin de garantir l'inclusion de tous les utilisateurs, y compris ceux qui souffrent de déficiences visuelles. Les contenus visuels actuels ne parviennent souvent pas à répondre aux besoins des utilisateurs de lecteurs d'écran en raison de l'absence de descriptions textuelles complètes. Pour combler cette lacune, nous proposons un cadre conçu pour permettre aux créateurs de contenus médiatiques de transformer les graphiques en récits descriptifs. Cet outil facilite non seulement la compréhension de données visuelles complexes par le biais du texte, mais favorise également une prise de sensibilisation à l'accessibilité dans la création de contenu numérique. Grâce à l'application de ce cadre, les utilisateurs peuvent interpréter et transmettre les informations des visualisations de données de manière plus efficace, en s'adaptant à un public diversifié. Nos évaluations révèlent que cet outil améliore non seulement la compréhension des visualisations de données, mais favorise également de nouvelles perspectives sur les données représentées, élargissant ainsi les possibilités d'interprétation pour tous les utilisateurs.

ABSTRACT

In the digital landscape, the ubiquity of data visualizations in media underscores the necessity for accessibility to ensure inclusivity for all users, including those with visual impairments. Current visual content often fails to cater to the needs of screen reader users due to the absence of comprehensive textual descriptions. To address this gap, we propose a framework designed to empower media content creators to transform charts into descriptive narratives. This tool not only facilitates the understanding of complex visual data through text but also fosters a broader awareness of accessibility in digital content creation. Through the application of this framework, users can interpret and convey the insights of data visualizations more effectively, accommodating a diverse audience. Our evaluations reveal that this tool not only enhances the comprehension of data visualizations but also promotes new perspectives on the represented data, thereby broadening the interpretative possibilities for all users.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii

RÉSUMÉ iv

ABSTRACT v

TABLE OF CONTENTS vi

LIST OF TABLES viii

LIST OF FIGURES ix

LIST OF SYMBOLS AND ACRONYMS x

LIST OF APPENDICES xi

CHAPTER 1 INTRODUCTION 1
- 1.1 Digital Accessibility 1
- 1.2 Screen Readers 2
- 1.3 Textual Descriptions 3
- 1.4 Motivation and Research Objectives 4
- 1.5 Contributions 5

CHAPTER 2 LITERATURE REVIEW 6
- 2.1 Existing Guidelines and Solutions 6
- 2.2 Tactile Graphics 7
- 2.3 Screen Reader Navigation 8
- 2.4 Visual Question Answering 9
- 2.5 Sonification 9
- 2.6 Textual Descriptions 11

CHAPTER 3 FRAMEWORK DESIGN 13
- 3.1 Design Challenges 13
- 3.2 Design Requirements 15
- 3.3 Workflow 16
  - 3.3.1 Chart Selection 17

3.3.2 Feature Identification and Selection . . . 17
3.3.3 Description Authoring . . . 20
3.4 Implementation . . . 20
3.4.1 Range of Compatible Chart Types . . . 21
3.4.2 Frontend . . . 22
3.4.3 Backend . . . 26

CHAPTER 4 EVALUATION . . . 36
4.1 Usage Scenarios . . . 36
4.1.1 Univariate Categorical Data . . . 37
4.1.2 Multivariate Categorical Data . . . 38
4.1.3 Multivariate Numerical Data . . . 39
4.2 User Study . . . 41
4.2.1 Qualitative Study . . . 41
4.2.2 Quantitative Study . . . 45

CHAPTER 5 CONCLUSION . . . 49
5.1 Summary of Works . . . 49
5.2 Limitations . . . 49
5.2.1 High Volume of Information . . . 50
5.2.2 Lack of Standardized Accessibility Guidelines . . . 51
5.2.3 Time-Saving Efficiency . . . 51
5.2.4 Data-Driven Narrative . . . 52
5.3 Future Research . . . 53

REFERENCES . . . 54

APPENDICES . . . 65

LIST OF TABLES

Table 3.1	Collection of available Datawrapper visualization types	22
Table 3.2	Extracted data	28
Table 3.3	Data structures supported by each chart type	33
Table 3.4	Example of a data frame	33
Table A.1	Feature-specific annotations based on chart type	65

LIST OF FIGURES

Figure 3.1 Main interface with three components: A. List of features in input chart, B. Generated descriptions of selected features, and C. Input chart 13
Figure 3.2 Workflow of the framework 17
Figure 3.3 Tree representation categorizing detected features on an input chart . 19
Figure 3.4 Number of visualizations published between 2021 and 2024 23
Figure 3.5 Login page 24
Figure 3.6 Import page 25
Figure 3.7 Samples page 26
Figure 3.8 Export page 27
Figure 3.9 Template for the general information section of a bar chart 31
Figure 3.10 Template for the data facts section of a bar chart 34
Figure 3.11 Template for the data facts section of a grouped bar chart 35
Figure 4.1 API access tokens in Datawrapper 36
Figure 4.2 Simple bar chart 37
Figure 4.3 Grouped column chart 38
Figure 4.4 Line chart 40
Figure 4.5 Survey scores 48

LIST OF SYMBOLS AND ACRONYMS

API	Application Programming Interface
ARIA	Accessible Rich Internet Applications
DenseNet	Dense Convolutional Network
DOM	Document Object Model
GIF	Graphics Interchange Format
HTML	HyperText Markup Language
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
LSTM	Long short-term memory
OCR	Optical Character Recognition
ResNet	Residual Neural Network
REST	Representational State Transfer
SVG	Scalable Vector Graphics
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VQA	visual question answering
WCAG	Web Content Accessibility Guidelines
W3C	World Wide Web Consortium
XML	Extensible Markup Language

LIST OF APPENDICES

Appendix A Annotation Techniques 65
Appendix B Interview Questions 66
Appendix C Online Survey 68

CHAPTER 1 INTRODUCTION

In our increasingly data-driven world, visualizations serve as pivotal tools for storytelling, offering clear and concise portrayals of complex datasets that cater to the swift comprehension needs of modern audiences [1]. These visual representations transform abstract numbers into insightful narratives and intuitive graphics, making complex information more accessible to the general public. However, an exclusive reliance on these visual formats inherently excludes significant segments of the audience, including those with visual impairments. Extensive research has demonstrated that a substantial number of online visualizations are not compatible with assistive technologies like screen readers, primarily due to the absence of adequate textual alternatives [2]. This limitation not only restricts the accessibility of such visualizations for visually impaired individuals and screen reader users but also affects those requiring auditory information for various activities, such as driving or multitasking. Consequently, to bridge this accessibility gap and ensure equitable access to information, the provision of textual descriptions for visualizations is not merely important—it is imperative.

1.1 Digital Accessibility

According to the World Health Organization, over 2.2 billion people worldwide live with some form of visual impairment, ranging from partial sight loss to complete blindness [3]. To support these individuals to access information on computers and mobile devices, a variety of assistive technologies have been developed. These innovative tools include screen magnifiers that enlarge on-screen text and images for those with low vision, screen readers that convert text into speech, braille translators that transform computer files into braille files, braille printers that receive data from computer devices and create tactile dots on special heavyweight paper, and refreshable braille displays that provide tactile reading and writing options by electronically raising and lowering pins. Among these options, screen readers are the primary tools for converting computing interfaces and digital content into non-visual formats. Their widespread use stems from several advantages. First, screen readers are cost-effective, with many options like NVDA (NonVisual Desktop Access) available for free or at a low cost. Second, they integrate seamlessly within computer systems, providing access to operating systems, applications, and web content. Third, as software-based solutions, they are portable, convenient, and offer real-time interaction with digital content. In professional environments, advanced vision-impaired computer users commonly use a braille display alongside an audio-based screen reader [4].

1.2 Screen Readers

Screen readers are software applications designed to convert text and other on-screen elements into spoken words or braille output using a refreshable braille display. Acting as interfaces between the computer's operating system, its applications, and users, screen readers facilitate interaction with digital content for individuals with visual impairments. When providing spoken word output, these applications employ text-to-speech technology through a speech synthesizer to audibly convey content displayed on the screen. Users primarily interact with screen readers through a series of keyboard shortcuts, enabling them to navigate through texts and execute commands. This interaction allows users to manage and operate their devices without the need to see the screen.

For web pages, screen readers work by interpreting the HyperText Markup Language (HTML) structure and attributes. When a web page is loaded, the software parses the HTML code, recognizing the roles, states, and properties of elements on the user interface according to the semantic HTML and Accessible Rich Internet Applications (ARIA) suite. Semantic HTML refers to the use of tags that describe their purpose to the browser, thus conveying the meaning of the content they contain rather than merely defining the presentation or layout of the page. For example, `header` and `footer` are semantic tags that clearly define the role and meaning of the enclosed content, while `div` and `span` are generic, non-semantic elements [5]. The ARIA suite, on the other hand, is a technical specification that provides a framework for adding attributes to dynamic content and advanced user interface controls, enabling better interaction and navigation for assistive technologies.

The effectiveness of screen readers is greatly enhanced by adhering to accessibility standards, such as those outlined in the Web Content Accessibility Guidelines (WCAG) developed by the World Wide Web Consortium (W3C). Proper implementation of semantic HTML and ARIA roles ensures that screen readers can accurately interpret and present web content to users with visual impairments. Key practices include labeling elements, providing descriptive alt-text for images, and ensuring that all interactive elements are keyboard accessible. Without these accessibility measures, users relying on screen readers may find it challenging to navigate and understand web content, leading to a frustrating experience.

On the market, several screen readers are available, each offering unique features tailored to different operating systems. Among the most popular are JAWS (Job Access With Speech), NVDA, and VoiceOver [6]. JAWS is one of the most widely used screen readers for Windows, known for its extensive features and customization options. It provides robust support for a wide range of applications and web content, making it a preferred choice for professional settings. NVDA, also for Windows, is remarkable for being a free and open-source alternative. Despite being cost-free, NVDA offers a comprehensive set of features and is highly regarded within the visually impaired community for its reliability and ongoing development. VoiceOver, the built-in screen reader for macOS and iOS, is seamlessly integrated with the Apple ecosystem, ensuring a consistent user experience across all Apple devices. It is particularly praised for its touchscreen gesture support on iOS, allowing users to navigate their devices using a series of touch gestures in addition to traditional keyboard shortcuts.

1.3 Textual Descriptions

Screen readers rely on HTML and ARIA to present information. When encountering a non-textual element, a screen reader searches for embedded information. For images, it reads the alt-text or long description if available. According to WCAG 2.0, the `alt` attribute for `img` elements should be a short text alternative that conveys the essential meaning of the image [7]. In contrast, the `longdesc` attribute should be a Uniform Resource Identifier (URI) pointing to a separate resource that offers a more comprehensive description, detailing the image's visual characteristics [8], which is particularly useful for images with complex visual information. While the use of alt-text has become widespread due to its simplicity, adherence to good coding practices, and advancements in front-end technologies, long descriptions are often entirely missing [9]. When they are provided, they can sometimes be vague or inaccurately describe the images.

Although web content writers often overlook the provision of long descriptions, the practice of describing non-visual elements through text or spoken words has a long history. One well-known practice is audio description, which originated in the 1970s and is primarily used in theater performances, film, video, and museum exhibitions [10]. Over decades, audio description has evolved into a standard feature across various forms of media, including book and press illustrations. This advancement is supported by well-established guidelines [11], university courses, and professional training programs [12]. Like any specialized skill, mastering audio description requires dedicated practice to develop essential competencies. Practitioners have to learn to observe and interpret visual details while knowing the context in which they are presented. They also need to develop linguistic skills and content-related expertise, including a thorough understanding of disability and accessibility issues. Moreover, technological proficiency in using any specialized audio description software is required, along with personal and general competencies like interpreting, analyzing, and summarizing information [13].

For websites and digital publications, the practice of providing detailed descriptions for non-textual elements is gaining traction; however, the lack of precise and standardized guidelines significantly hampers consistency in these efforts [14]. In a series of interviews conducted by Kasdorf with leading publishers and service providers, it was revealed that image descriptions are the most common missing feature in publications [15]. Publishers identified this as the most significant challenge across all types of content. One noted that even experienced vendors struggle with the volume of work and often produce poor-quality descriptions that require rework. While adding image descriptions currently demands extra effort, it is anticipated that this will become a standard requirement for publication manuscripts in the future.

1.4 Motivation and Research Objectives

Ensuring equal access to information and communication technologies is crucial, as digital accessibility is not only a matter of convenience but also a fundamental right. Accessibility in digital content guarantees that websites, publications, and applications are usable by everyone. This inclusivity is essential for providing fair opportunities in education, employment, social interaction, and access to services. Furthermore, adopting digital accessibility best practices offers benefits beyond assisting those with disabilities. For example, implementing features such as long image descriptions can enable sighted users engaged in other activities to listen to information. Technically, these practices also enhance search engine optimization by making content more understandable and indexable. Thus, digital accessibility not only promotes equality but also improves overall usability and information visibility.

This project is conducted in partnership with industrial collaborators Le Devoir and Radio Canada, with the goal of enhancing the accessibility of online visualizations in news articles. Although providing long descriptions is part of the WCAG, the task of effectively describing visual elements in data-driven stories introduces multiple challenges. First, there is a notable learning curve associated with the skill of writing comprehensive descriptions for accessibility [13], as it requires an in-depth understanding of the critical details to be included. Second, data visualizations present complex datasets that demand advanced analytical skills [16]. Creators must interpret visual patterns with precision, ensuring that the data is neither oversimplified nor misrepresented. Third, crafting such descriptions is often laborious and time-consuming, necessitating meticulous attention to detail. Despite diligent efforts, some descriptions may not adequately convey all the necessary details of the visual elements [17]. Recently, the machine learning community has explored deep learning-based methods to aid chart-text transformation [18–21]. Yet, these black box models often lack transparency, challenging users' ability to fine-tune or fully comprehend the details in the generated outputs.

To further investigate the challenges, we consulted a few journalists from the industrial partners. These discussions confirmed that they use Datawrapper to design visualizations for news articles. Datawrapper is a user-friendly online tool that allows users to make charts, maps, and tables without the the need for programming or coding skills. Although both Datawrapper and the companies' content management systems provide fields for long image descriptions, journalists typically only include the title as the sole textual addition within these visualizations.

In response to these issues, we propose the development of a novel framework designed to aid media content creators in interpreting charts and producing detailed textual descriptions. This preliminary research is tailored specifically for those who develop visualizations using Datawrapper and have basic proficiency in statistics and data analysis.

1.5 Contributions

The contributions are organized by chapter, as outlined below:

- **Chapter 2** explores existing solutions, presenting previous related works to establish a solid foundation for understanding the context within the broader academic discourse.
- **Chapter 3** presents the design of our solution, a novel heuristic approach that affords authors greater control over the process of authoring descriptions, enhancing the precision and relevance of the output. Additionally, the chapter details the implementation of the solution, elaborating on the technical procedures involved in the automatic method for identifying features in visualizations, which streamlines different stages of the description process by highlighting critical data elements automatically.
- **Chapter 4** discusses a comprehensive user study to evaluate the practical utility of our framework. The study not only validates the effectiveness of our method but also highlights areas requiring further exploration to enhance the framework's applicability and functionality.

CHAPTER 2 LITERATURE REVIEW

Making online visualizations accessible involves translating visual information into alternative non-visual formats. The topic has received considerable attention and interest across multiple fields, particularly in the areas of accessibility, human-computer interaction, computer vision, and natural language processing. Experts in these disciplines explore innovative methods to enhance the accessibility of complex visual content. These efforts ensure that information reaches a broader audience, promoting inclusivity and diversity.

2.1 Existing Guidelines and Solutions

Digital accessibility ensures that all individuals can access, navigate, and interact with digital content. This encompasses a wide range of content types, including websites, mobile applications, and e-books. The W3C plays a crucial role in establishing accessibility guidelines and standards for both web content and electronic publications. The WCAG, recognized by International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), provide a conceptual framework structured around four key principles [22]:

1. **Perceivable**: Information and user interface elements should be presented in ways that are visible or detectable to all users. This includes providing text alternatives for non-text content to ensure that the content can be adapted to different formats, allowing users to process the information.

2. **Operable**: Users should be able to interact with all controls and interactive elements, and navigate the interface using various input methods, including computer peripherals and assistive technologies. This principle also emphasizes the importance of giving users adequate time to engage with content and avoiding using flashing elements that could potentially trigger seizures or photosensitive reactions.

3. **Understandable**: User interface information and navigation should be clear and consistent. This means ensuring text content is readable and easy to understand, and that web pages behave in predictable ways, helping users avoid errors during interaction.

4. **Robust**: Content should be consistently accessible and interpretable by a wide range of user agents, including assistive technologies. Web pages should be compatible with different browsers, platforms, and devices, ensuring that the content remains functional as technologies evolve.

While the WCAG outline the principles and guidelines for web accessibility, ARIA provide technical solutions and specifications. ARIA define a set of attributes that can be added to HTML elements to enable assistive technologies to interpret and navigate web content. These attributes are used to achieve accessibility standards, especially for dynamic and interactive elements, when traditional HTML falls short in capturing the complexity and interactivity of modern web applications. Visualization libraries such as Vega-Lite [23] and Highcharts [24] employ these techniques to ensure compatibility with screen readers. Concurrently, ongoing research explores different principles [25] and design dimensions to improve the accessibility of online visualizations [26, 27]. These dimensions consist of novel approaches to navigating, comprehending, and interacting with graphical data using non-visual interfaces that rely on touch and auditory feedback instead of sight. Touch-based methods primarily involve tactile graphics, which provide raised representations of visual images that can be understood through touch. Auditory methods address both interaction and information translation, including screen reader navigation, visual question answering (VQA), sonification, and textual descriptions. The following sections will discuss the state-of-the-art approaches for each of these solutions.

2.2 Tactile Graphics

Tactile graphics are raised representations of visual images designed to convey information through touch. These graphics are produced using a variety of techniques and materials, such as thermoforming, swell paper, and embossing on thicker paper. Experts indicate that preparing and completing a tactile graphic can take several hours. Although image manipulation applications like Adobe Photoshop have slightly decreased this time frame [28], creating tactile graphics continues to be a labor-intensive process. Additionally, the production of tactile graphics is hindered by inadequate funding for braille production and a shortage of braille transcribers [29].

Recent research has concentrated on streamlining the process of creating tactile visualizations. Innovations in this area primarily involve the development of automatic tactile graphics generation methods leveraging computer vision and machine learning techniques [30]. For instance, a tactile graphics assistant was developed to enhance the efficiency of specialists in creating these graphics by utilizing image processing and classification algorithms, and it was tested in the field with practitioners [31, 32]. Brown and Hurst created VizTouch, a software program that generates tactile visualization files ready to be manufactured on a 3D printer [33]. Similarly, Štampach and Mulíčková developed a technological procedure in Python scripts for the preparation of tactile maps for ArcGis environment [34]. Recently, Guinness et al. explored dynamic tactile graphics by developing RoboGraphics, a prototype system combining a touchscreen tablet, static tactile overlays, and small mobile robots [35].

Despite the benefits of tactile graphics, they have some limitations. A key drawback is their limited ability to handle extensive information, as the fingers' capacity to distinguish details is significantly less precise compared to visual acuity [36]. Consequently, the average amount of information on a tactile graphic is lower than that on an equivalent classical image. To address this issue, researchers have been exploring alternative techniques such as sound and haptics to enhance tactile graphics with supplementary information. Sound is mainly used to provide text-to-speech information that serves as explanations through object and text recognition [37–39], while haptic-tactile graphs increase interactivity by producing feedback based on user contact [40–42].

2.3 Screen Reader Navigation

For assistive technologies that rely on auditory output, screen readers are the most commonly used tools to read web pages. While modern web pages use spatial arrangement and typographic elements to present semantic relationships and complex information, screen readers typically overlook this rich visual structure and render the content as a linear, one-dimensional sequence of spoken words. As a result, much of the intended visual semantics can be lost. For visually impaired users, navigating this linear content can be extremely tedious and time-consuming compared to sighted users who can quickly scan a page and grasp its key points at a glance [43].

Recent studies on visual semantic understanding among visually impaired users of laptops and smartphones have demonstrated that access to visual semantics can enhance collaboration with sighted users and facilitate navigation and interaction with user interface elements [44]. One approach to improve the screen reader experience consists of restructuring web content into more accessible formats. For example, Williams et al. conducted experiments with screen reader users to evaluate the effectiveness of transcoding grid-based layouts of web pages into tables. Their findings revealed that tables not only facilitate navigation but also enable more accurate data lookup [45]. Following the same principle, Zong et al. proposed a navigation structure that improves the user experience when interacting with online visualizations by creating a logical and predictable reading order, ensuring that visually impaired users can navigate and interpret data visualizations more effectively [46].

Given a visualization, it is important to recognize that while certain individuals prefer direct interaction with or access to the underlying data [47], focusing solely on raw data can be limiting. To accommodate individual preferences, visualization libraries like Google Charts [48] make data tables accessible by default. However, visualizations offer a holistic overview that allows sighted viewers to intuitively and efficiently grasp insights, patterns, and trends which would be more challenging to discern from data points alone. The graphical representation aids in identifying outliers, anomalies, and correlations that might be missed when inspecting data in a tabular format. Therefore, while access to raw data is valuable, visualizations play a crucial role in presenting a thorough and clear summary of the information.

2.4 Visual Question Answering

Another approach for accessing information within images is VQA, which refers to the task in which a system is presented with an image along with a textual query, with the objective of generating an accurate natural language response to the question. Inspired by this concept, datasets like FigureQA [49] and DVQA [50] which stands for data visualization question answering were created. These datasets consist of images, questions, and answers about scientific figures and bar charts. Based on these datasets, Kafle et al. proposed a novel algorithm for question answering that employs Long short-term memory (LSTM) and Dense Convolutional Network (DenseNet), along with an additional Optical Character Recognition (OCR) component specifically for DVQA [51]. On the other hand, visualization datasets can take various forms, including static images, JavaScript code, Scalable Vector Graphics (SVG), and Canvas. Among these, SVG is particularly advantageous for adding rich coded semantic and contextual information [52]. For JavaScript-based input, Kim et al. developed an automatic chart question answering pipeline that extracts references to visual attributes from natural language questions and transforms them into references to data [53]. To facilitate interaction with screen readers, Sharif et al. created a speech recognition plug-in and integrated it into Google Charts, D3, and ChartJS [54]. Question answering in visualizations allows individuals to interact with visual data through natural language queries to quickly extract specific information. However, the accuracy of VQA systems depends on several factors, such as the quality of training data and the ambiguity within the queries. Moreover, without offering any initial information, individuals may lack the necessary context to formulate appropriate questions.

2.5 Sonification

Sonification refers to the use of non-speech audio to facilitate communication and interpretation of information. The research communities for visualization and sonification share similar goals, with the key difference being that the latter focuses on acoustic signals rather than visual data. This field of study is driven by the necessity to understand large datasets and the growing availability of advanced media technologies [55]. Research in auditory perception has demonstrated that humans are highly sensitive to temporal characteristics [56, 57], enabling them to distinguish between periodic and aperiodic events and to detect subtle changes in the frequency of continuous signals. As a result, sonification is particularly advantageous for understanding or monitoring complex temporal data.

Sonification applications and designs have been developed based on multimodal perception across various scientific disciplines. Notable examples include the Geiger counter, which detects invisible radiation and alerts users, and the pulse oximeter, which emits a tone that changes pitch based on the oxygen level in a patient's blood, allowing medical professionals to monitor crucial information. There are five main techniques in sonification. Audification is a technique that represents long sequences of data values by converting them into digital audio waveforms characterized by variations in frequency and loudness. Parameter mapping sonification translates data values into auditory channels like pitch, timbre, and loudness. Model-based sonification involves creating a dynamic model to describe changes in a system over time. Auditory icons are short, distinctive sounds that closely mimic real-world noises, serving a similar function to visual icons. Lastly, earcons, comparable to visual symbols, are used to represent specific events. Recent research combining sonification and visualization mainly employs parameter mapping sonification [58]. From an accessibility perspective, some sonification applications and designs are created specifically for visually impaired individuals. For instance, Apple's audio graphs API provides information to VoiceOver, converting data into sound by encoding each axis as audio [59]. Similarly, Hightcharts developed Sonification Studio, a multimodal graphing platform for auditory display, which was tested by both sighted and visually impaired participants [60].

Sonification has several advantages, such as the ability to convey time series data through an additional sensory channel, which is particularly useful in environments where visual attention is limited. It also delivers real-time feedback and can be integrated into applications to enhance data understanding and monitoring. However, not all datasets are suited for auditory representation. Additionally, while learning to recognize auditory patterns can significantly improve the efficiency of data interpretation, certain applications necessitate specific training [61].

2.6 Textual Descriptions

When it comes to translating information, one of the most effective methods to make online visualizations accessible is through the use of textual descriptions. Unlike tactile graphics, which require additional hardware, and sonification, which is primarily effective for time series data, textual descriptions provide written insights into the content of images. They capture essential elements such as objects, colors, and spatial relationships, allowing individuals who cannot see the images to form a mental picture and understand the content through descriptive texts. This aligns with the recommendations from WCAG, which advocate for text alternatives for all non-textual web content, ensuring that these alternatives fulfill the same function. The W3C particularly stresses the importance of crafting detailed descriptions for complex images such as charts, maps, and other data visualizations. The process of converting these complex images into text involves a substantial reduction in information dimension, which brings forth critical considerations regarding the specific needs and preferences of the target audience. Consequently, recent studies increasingly aim to pinpoint what aspects of these descriptions are most valued by visually impaired users [47, 62]. This approach ensures that the descriptions meet the practical needs of those who rely on them, while other strategies focus on facilitating the collection and the generation of textual descriptions, and the creation of figure captioning datasets like LineCap [63].

For collecting descriptions, certain browser extensions have been developed. Examples include Caption Crawler, which utilizes reverse image search to locate existing captions on the web and apply them to images without captions on other websites [64]. Another extension, Twitter A11y, enhances accessibility on Twitter by adding alternative text to images. It leverages external links, OCR, screenshots of retweets, reverse image search, automatic image captioning, and crowdsourcing to provide these descriptions [65]. Additionally, Ga11y, an automated Graphics Interchange Format (GIF) annotation system, combines crowdsourcing and machine-generated annotations to ensure that images in GIF are accessible to all users [66].

Regarding the automatic generation of text from visualizations and structured data, significant advancements in deep learning have been made. Chen et al. employed Residual Neural Network (ResNet) and LSTM architectures to generate natural language descriptions for figures in the FigCAP dataset [18], while Cheng et al. proposed a framework with transformer-based chart detection and pre-trained vision-language model [21]. Additionally, some researchers prioritize the underlying data rather than the visual representation of charts, focusing on chart-to-data [67] and data-to-text generation. Notable efforts in the latter area include the use of transformer-based models [19, 68] and encoder-decoder LSTM frameworks for analyzing time-series data [69]. In assessing the performance of these models, the BLEU [70] and ROUGE [71] metrics are commonly applied to test datasets. These advancements highlight the growing sophistication of automatic text generation techniques, although challenges in generalization and credibility remain areas for further research and development. To tackle issues in natural language generation, some research prototypes [72, 73] and language generation methods [74] have adopted template-based text generation techniques, which offer more transparent and reliable approaches to creating textual descriptions from visual data.

While significant advancements have been made in the automatic generation of text from visualizations and structured data, several limitations persist. Deep learning models, despite their computational skills, often struggle to generalize well across different datasets, limiting their practical use with unfamiliar datasets in real-world scenarios. Moreover, these models have an opaque nature, which obscures the internal mechanisms and decision-making processes, raising credibility concerns. This lack of transparency can undermine user trust, particularly when the generated descriptions are incorrect or misleading. Thus, incorporating a human-in-the-loop approach is crucial in addressing these limitations. By involving human users in the generation process, a system can leverage human expertise to validate and refine the outputs, ensuring higher accuracy and reliability. This approach not only enhances the quality of the generated texts but also increases transparency and user trust, making it a more robust solution for real-world applications where nuanced interpretation and accountability are essential.

CHAPTER 3 FRAMEWORK DESIGN

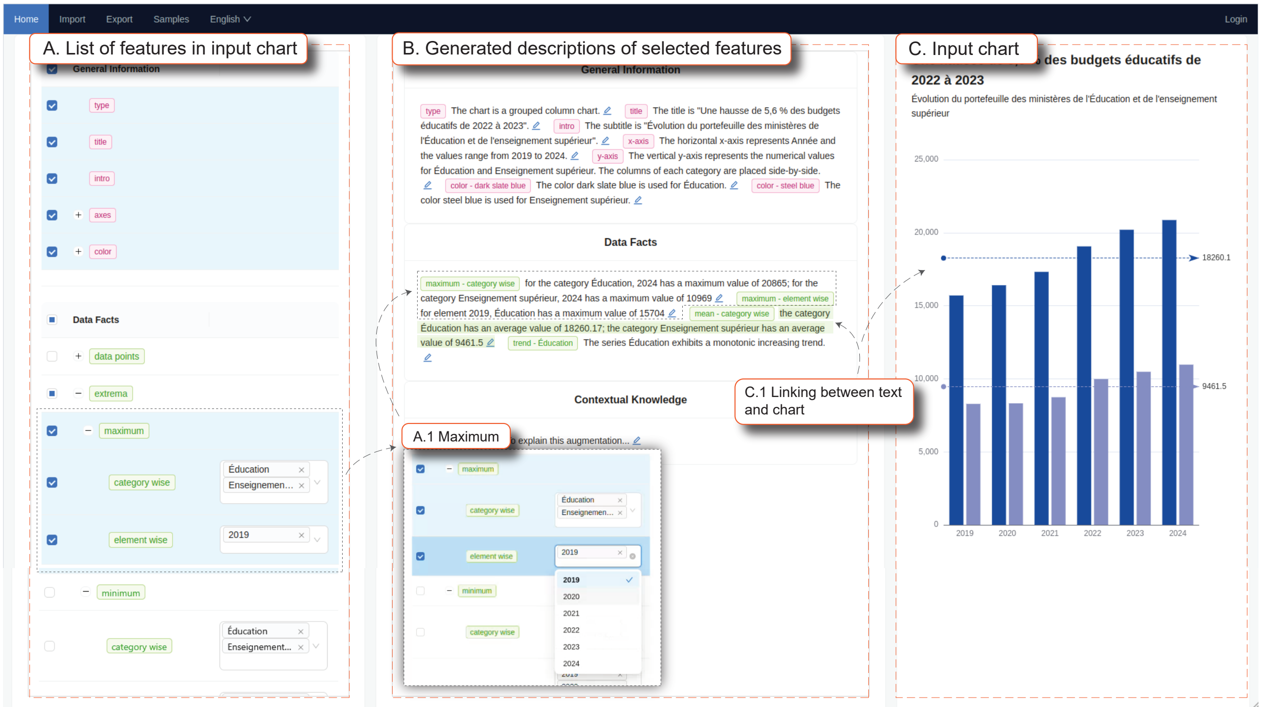

Figure 3.1 Main interface with three components: A. List of features in input chart, B. Generated descriptions of selected features, and C. Input chart

To convert a chart to text, automatic methods that employ deep learning make it hard for users to integrate their contextual information. Moreover, these methodologies typically fall short in considering the precise content requirements necessary for meaningful descriptions, which also necessitate an accurate interpretation of the data. Such shortcomings prompt the need for alternative strategies. As a result, we decided to create a methodology that facilitates the exploration of data to refine automatically generated descriptions by adopting a heuristic method that incorporates a human-in-the-loop approach. Such an approach allows human expertise to improve the accuracy and relevance of the textual outputs but also ensures that the descriptions are appropriate and complete.

3.1 Design Challenges

We distill four design challenges (**C1-C4**) from the literature review and discussions with experts in communication and accessibility.

C1. **A design that does not fit well with existing systems and processes can cause difficulties for users to adopt it.** When a new design is introduced, users often face a steep learning curve, requiring them to invest considerable time and effort to understand it, leading to an increased cognitive load. This additional mental burden can generate frustration, especially if the design is complex or significantly different from existing systems and processes. Such frustration can turn into resistance to change, primarily driven by uncertainty, which is often related to concerns about how the design will fit into existing workflow and whether it will disrupt established routines. Compatibility issues play a central role here [75]. When the design is not fully compatible with the current workflow, it can create disruptions that hinder productivity and generate further resistance, as users may fear that it will require them to abandon familiar practices or adopt new ones that are less effective, leading to a potential decline in performance or satisfaction. To ensure successful adoption, it is important to create solutions that are closely aligned with users' existing practices, ensuring compatibility with current systems and processes to minimize disruptions and ease the transition.

C2. **Properly describing a visualization requires a thorough examination of all its features to ensure that the message is accurately conveyed and accessible to all users.** This process involves more than simply explaining what it depicts. It demands a deep understanding of accessibility principles and a keen awareness of the information needs of the target audience. The accessibility principles are essential for making the visualization usable and meaningful to a diverse range of people, including those with varying levels of familiarity with the subject matter and different degrees of statistical expertise. In practice, this means that the description of a visualization should consider several critical elements. These include the structural layout, which involves the arrangement of data and the flow of information; labeling, which ensures that all data points and axes are identified; and the use of patterns, textures, or color contrasts to differentiate between data points. Additionally, the content of the data itself must be carefully considered to ensure that it is presented in a way that is both accurate and easy to understand [62]. The description should account for how these features include sufficient details to enable readers to grasp the overall structure and key insights of the visualization without direct visual access [47], particularly for those who rely on screen readers or other assistive technologies.

C3. **The accuracy of the description depends on the correct interpretation of the visual elements and the underlying data.** Fully automated systems, while increasingly sophisticated, can still make critical errors [76], as their performance is closely tied to the quality and scope of the training data. If the training data is biased, incomplete, or not representative of real-world scenarios, these systems may struggle to accurately interpret and describe situations that deviate from the learned patterns, leading to inaccurate or misleading results. This becomes especially problematic for users who rely entirely on these descriptions to form mental models of the content, as they may be unable to cross-check or verify the information independently. Errors in interpretation can result in significant misunderstandings and potentially lead to misinformed decisions, particularly in high-stakes environments where accuracy is paramount. Therefore, human expertise remains crucial in reviewing and refining these descriptions to ensure they are both accurate and contextually appropriate, thereby mitigating the risks associated with automated systems and ensuring reliable outcomes.

C4. **Users may be resistant to fully trusting auto-generated outputs, especially when the steps taken to generate those outputs are not clearly shown.** When visualization is translated into text, users might question whether the resulting text truly captures all the nuances and critical details of the visual content. This uncertainty often compels them to cross-check the text against the original visualization to ensure that the key details and insights are correctly represented [77]. The need for such verification demands time and cognitive effort, which can diminish the perceived convenience of the auto-generated output. Instead of saving time, users may find themselves investing additional mental energy to ensure that the transformation from visual to text does not lead to any loss or misinterpretation of information. This extra step reduces the overall value of automated processes and reflects a lack of trust. Users are often hesitant to rely entirely on automated processes because they cannot be certain that the output is accurate without their intervention, highlighting the ongoing need for manual verification and the importance of transparency in automation.

3.2 Design Requirements

Based on the challenges, we derive four requirements (**R1-R4**) to guide the design of our framework.

R1. **The design should be compatible with the existing workflow to ensure high usability and a smooth adoption process.** For media content creators, it is crucial that any new design integrates seamlessly with the software and processes they rely on. This compatibility not only minimizes the learning curve but also enhances usability, enabling target users to quickly adapt without the need to alter their established methods. This reduces operational disruptions and ensures that users can continue to work efficiently using the tools they are comfortable with.

R2. **The framework should extract features from the input visualization and present them in a structured format to facilitate a clear understanding of the visualization's composition.** Studies have shown that authoring interfaces that guide authors in determining what content to include in their descriptions lead to higher-quality descriptions [78], which further supports the need for a structured approach. These features should be identified and verified through a heuristic approach to ensure accuracy and relevance. By organizing the features in a structured manner, the framework helps users categorize complex information, thereby enhancing the efficiency of data analysis. This organized presentation also reduces cognitive load, making it easier for users to understand the chart's components.

R3. **Each feature needs to have a corresponding description to ensure that the information is fully accessible and understandable to the user.** These generated descriptions must be accurate to correctly reflect the nuances and details of each feature. Additionally, they should be editable and rearrangeable, allowing users to refine, adjust, and organize the text to better suit their specific needs or interpretations in a way that best supports their comprehension. This flexibility in writing management enhances the overall usability of the framework, making it more adaptable to diverse user requirements and contexts.

R4. **Connecting the generated descriptions to the original visualization through linking and brushing techniques for maintaining a clear and intuitive relationship between the output and the original input.** These techniques allow users to see real-time updates on the chart as they interact with the descriptions, allowing for quick verification and understanding. To enhance this process, the design should incorporate visual cues to illustrate the meanings of the descriptions, drawing attention to key points and guiding the viewer's focus to the most relevant part of the chart. This helps users to more easily comprehend and trust the connections between the descriptions and the chart, ultimately improving the effectiveness of description authoring.

3.3 Workflow

Our design and implementation process is guided by literature reviews from visualization and accessibility research. The workflow of our framework is designed to streamline the conversion

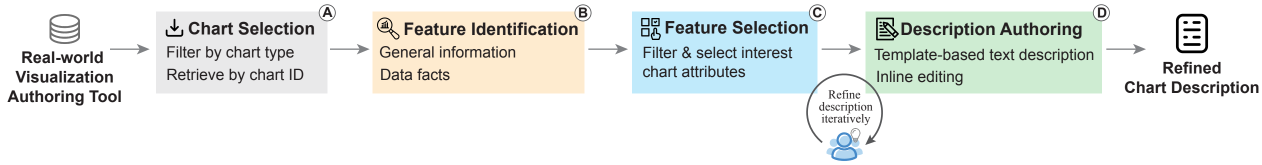

Figure 3.2 Workflow of the framework

of an input chart into a textual description. Initially, users input a chart (Figure 3.2A), prompting the framework to extract both data and metadata and to identify key features essential for detailed analysis (Figure 3.2B). This marks the beginning of an interactive phase, where users play an important role. In this phase, users critically assess the extracted features, select those most relevant, and meticulously refine the descriptions for each (Figure 3.2C and Figure 3.2D). This cycle of selection and refinement is iteratively repeated as necessary to enhance the narrative and accuracy of the data's story. In the following section, we will discuss each of the four steps of the workflow presented in Figure 3.2.

3.3.1 Chart Selection

In the initial phase, users are required to give an input chart, which serves as the starting point for the workflow. The framework is specifically designed to integrate with the Datawrapper database, a widely recognized visualization tool used by media content creators for data-driven storytelling. During the selection step, users can filter charts based on type, a feature beneficial for those who need to quickly narrow down relevant visualizations from a large collection. Additionally, the framework allows users to retrieve charts using their unique IDs, allowing direct access to specific ones within the database.

3.3.2 Feature Identification and Selection

The identified features are displayed in an interactive list (Figure 3.1A), facilitating a comprehensive understanding of the elements comprising the input chart. The process of feature identification and selection is structured into three distinct phases: **1) data extraction**, where relevant data are retrieved from the input; **2) feature detection**, during which the framework identifies key elements necessary for analysis; and **3) feature presentation and selection**, where these elements are displayed for user interaction and selection.

Data Extraction. Given an input chart, the framework extracts raw data and metadata by connecting to the database. The raw data, presented in tabular format, refer to the foundational elements that construct the chart, such as numerical and categorical values. Metadata comprises a variety of details, including the chart's title, its specific type, and any accompanying notes. Additionally, it captures aesthetic aspects like color schemes and functional attributes such as whether the data is sorted and its sorting order.

Feature Detection. The framework conducts an automatic analysis of the extracted data and metadata to identify key features. The detection is based on chart type and currently supports all basic statistical charts and their variants such as bar charts (split bars, stacked bars, grouped bars, etc.), area charts, line charts, and pie charts. For each type of chart, the framework detects key features and classifies them into the following categories: general information and data facts [62].

For general information, the framework captures important details such as chart type, main title, subtitle, footnote, axes, and color schemes. The comprehensive understanding of visualization goes beyond mere data representation; it necessitates an elaborate description of all graphical elements. These detailed descriptions are crucial for visually impaired people, as they enable them to form a mental representation of the visualization, enhancing their comprehension of the overall design [47].

Regarding data facts, the framework analyzes the numerical and statistical elements of the chart. It identifies specific data points and calculates statistical measures such as extrema, mean, standard deviation, and median. Additionally, it recognizes outliers—data points that markedly differ from the main observations. When the chart involves numerical data on the independent axis, the framework assesses correlations and trends. These facts are derived from the low-level analytical tasks that viewers typically perform while engaging with a visualization [79]. These tasks include retrieving specific values, often serving as the foundation for deeper analysis; filtering to focus on particular subsets that meet specific criteria; computing derived value, such as averages; finding extremum, which helps in understanding the boundaries or limits of the dataset; sorting, which organizes data in a particular order; determining the range for identifying the span of data; characterizing distribution for assessing how data points are spread; finding anomalies for uncovering potential errors, outliers, or significant events; clustering, which allows users to group similar data points; and identifying correlations, which involves analyzing relationships between variables to determine dependencies.

In addition to those two categories, we have also incorporated a section for contextual knowledge, which allows users to input information explaining the purpose behind including the chart. A schema of detected features is illustrated in Figure 3.3. Depending on the chart type and the specific characteristics of the input, certain elements may be omitted or expanded as features and tasks a chart supports can vary depending on the dataset, the visualization, and specific design variations [80, 81]. For example, a pie chart does not have axes but follows a specific order for its slices. On the other hand, in a grouped bar chart, extreme and average values can be identified based on items within and across groups. The framework adapts to the unique attributes of each chart type, providing tailored insights and facilitating more precise data interpretation.

```
Input Chart
├── General Information
│   ├── type
│   ├── title
│   ├── subtitle
│   ├── footnote
│   ├── axes
│   │   ├── x-axis
│   │   └── y-axis
│   ├── sorting order
│   └── color
├── Data Facts
│   ├── data points
│   ├── extrema
│   │   ├── maximum
│   │   └── minimum
│   ├── distribution
│   │   ├── mean
│   │   ├── standard deviation
│   │   └── median
│   ├── outliers
│   ├── correlation
│   └── trend
└── Contextual Knowledge
    └── external information
```

Figure 3.3 Tree representation categorizing detected features on an input chart

Feature Presentation and Selection. The detected features are displayed as checkbox lists, as illustrated in Figure 3.1A. To enhance user understanding, elements within each category are color-coded to signify their semantic significance—for instance, pink for general information and green for data facts. For visualizations such as bar and column charts that exhibit univariate data, the features are presented in a straightforward series of checkboxes. Conversely, for more complex, multivariate visualizations like grouped column charts, the interface incorporates an additional dropdown list (Figure 3.1A.1). This feature allows users to select specific variables, thereby accommodating the intricacies of multivariate data analysis.

3.3.3 Description Authoring

Upon the selection of a checkbox, the framework initiates template-based text generation to fill the description component as shown in Figure 3.1B. Each segment of the generated description is associated with the feature's name and is color-coordinated with the corresponding checkbox for clarity. The framework employs heuristic analysis of the features within the input chart, where each specific condition is connected to a designated text template. In charts depicting univariate data, selecting a checkbox automatically generates a preset textual description. Conversely, in charts with multivariate data, the text is dynamically updated based on the selected variables, enabling detailed comparisons both within and between groups [82]. This functionality ensures that the displayed information is tailored to the specific categories, offering personalized and pertinent descriptions that align with the user's comprehension and interaction with the data.

To ensure that the generated text meets high standards of clarity and stylistic quality, the framework allows users to fine-tune descriptions. Users can edit or rearrange text segments effortlessly by dragging the associated tags. Furthermore, when users hover their cursor over any part of the description, an animation activates on the corresponding section of the input chart (Figure 3.1C). This animation, combined with visual cues, emphasizes the relevance of the text, offering a dynamic and interactive method for illustrating the relationship between textual descriptions and visual data [83, 84].

3.4 Implementation

The framework is developed as a web-based application because of its numerous advantages over traditional software, such as accessibility from any device with an internet connection, no installation required, and a consistent user experience across platforms. The application is separated into frontend and backend. The frontend is responsible for managing user interactions, while the backend retrieves and processes data from the Datawrapper database. They communicate through Representational State Transfer (REST) Application Programming Interface (API), utilizing JavaScript Object Notation (JSON) format for data exchange.

The frontend is built using React [85], a JavaScript library designed for developing dynamic and responsive user interfaces, particularly for single-page applications. React leverages the Virtual DOM to increase performance. Traditionally, the Document Object Model (DOM), which represents the structure of a web page, is directly updated. This is an expensive operation because it requires re-rendering parts of the page, which can be slow. The Virtual DOM, on the other hand, is a lightweight, in-memory representation of the actual DOM. When a component's state changes, React updates the Virtual DOM first. It then uses a reconciliation process to compare the new Virtual DOM with the previous version. This allows it to determine the minimal set of changes needed, reducing the number of direct manipulations and boosting performance. Additionally, React's component-based architecture promotes code reuse, enabling developers to efficiently manage and maintain user interface elements across the application.

On the backend, the application is powered by Flask [86], a lightweight Python web framework responsible for managing server-side logic. The backend handles heavy computational tasks, as it has direct access to server resources to perform intensive tasks without burdening the client side. In this application, the backend is tasked with querying the database, cleaning and processing the data according to specific conditions, generating textual outputs using predefined templates, as well as creating visualizations in JSON format, which are then transmitted and interpreted by a frontend visualization package.

3.4.1 Range of Compatible Chart Types

Datawrapper offers support for a wide range of visualizations, including charts, maps, and tables, each designed to meet different data presentation needs [87]. The collection of various visualization types enables insights from different datasets to be conveyed effectively, using the most appropriate method for each context. For instance, line charts are ideal for showing trends and comparisons over time, while maps are used to represent geographical data, highlighting spatial relationships. Table 3.1 shows the specific types supported by Datawrapper[1].

To narrow the project scope and identify the visualization types the framework needs to support, we began by analyzing the datasets from the accounts of Le Devoir and Radio Canada. We used the API `https://api.datawrapper.de/v3/charts`[2] to retrieve metadata for all visualizations in both accounts. This process involved recursively fetching data with the default `limit` query parameter, which defines the maximum number of items to fetch in each step. The response comprises a list of JSON objects that provide detailed information and settings for each visualization. We then merged the two datasets, filtered the resulting

[1] The documentation contains inconsistencies: visualizations with ID multiple-lines do not exist, and the ID d3-bars-grouped is missing. The table has been updated to reflect the available datasets.

[2] `https://developer.datawrapper.de/reference/getcharts`

Table 3.1 Collection of available Datawrapper visualization types

Visualization type	ID
Bar Chart	`d3-bars`
Split Bars	`d3-bars-split`
Stacked Bars	`d3-bars-stacked`
Grouped Bars	`d3-bars-grouped`
Bullet Bars	`d3-bars-bullet`
Dot Plot	`d3-dot-plot`
Range Plot	`d3-range-plot`
Arrow Plot	`d3-arrow-plot`
Column Chart	`column-chart`
Grouped Column Chart	`grouped-column-chart`
Stacked Column Chart	`stacked-column-chart`
Area Chart	`d3-area`
Line Chart	`d3-lines`
Pie Chart	`d3-pies`
Donut Chart	`d3-donuts`
Multiple Pies	`d3-multiple-pies`
Multiple Donuts	`d3-multiple-donuts`
Scatter Plot	`d3-scatter-plot`
Election Donut	`election-donut-chart`
Table	`tables`
Choropleth Map	`d3-maps-choropleth`
Symbol Map	`d3-maps-symbols`
Locator Map	`locator-map`

data by verifying if the value of the key `publishedAt` in top-level properties falls between January 1st, 2021 and January 1st, 2024, and retrieved their corresponding type. Figure 3.4 shows the total number of visualizations published by the two companies over the past three years, categorized by type and sorted in descending order. We excluded maps and tables from our analysis and focused on supporting the ten most frequently used chart types. The supported charts, listed from most to least used, include: line chart, bar chart, column chart, grouped column chart, stacked bars, split bars, grouped bars, pie chart, area chart, and stacked column chart.

3.4.2 Frontend

The frontend is implemented as a single-page application featuring five main views, accessible via the navigation bar shown in Figure 3.1. These views include Home, Import, Export, and Samples on the left, and Login positioned on the far right. The navigation bar also includes

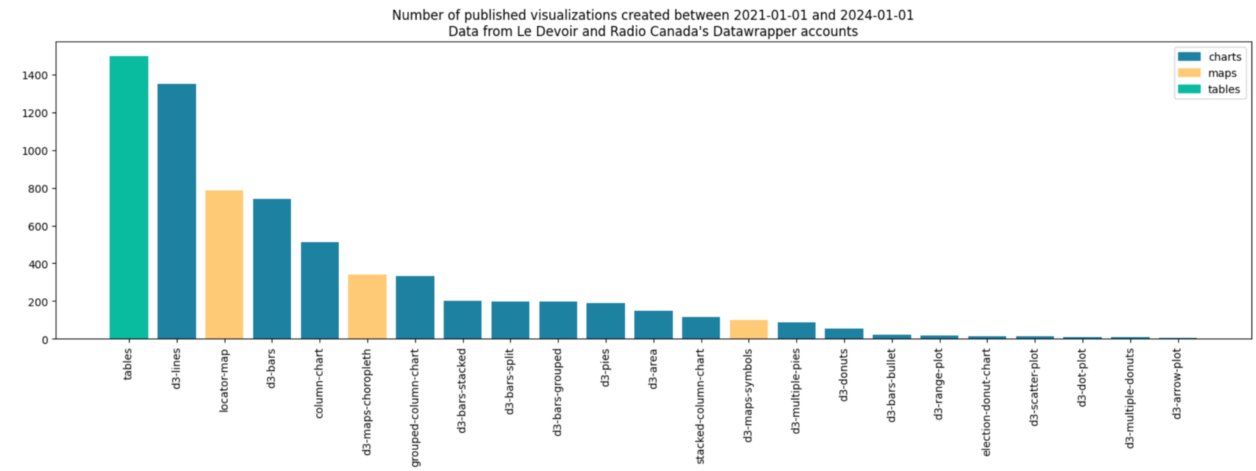

Figure 3.4 Number of visualizations published between 2021 and 2024

a language option, which is used to localize content in either English or French. Navigation between these pages is managed by a router that dynamically renders the appropriate components based on the Uniform Resource Locator (URL) path, eliminating the need for a full page reload. The language setting is handled using React Context, a mechanism that allows data to be shared across components without the necessity to explicitly pass it through each level of the component tree. For each language option, predefined terms and phrases are stored in a JSON file, ensuring low coupling and easy maintenance. The following section will provide an overview of each page within the frontend, highlighting their functionalities in the order of their respective workflow steps.

Login Page

To initiate the use of a chart within the framework, users must first generate a token in their Datawrapper account. The Login page (Figure 3.5) allows users to submit this token to the backend, where it is used to authenticate with the Datawrapper database through Bearer authentication. The backend calls the `charts` API and returns the status code to the frontend. A status code of 200 confirms successful authentication, granting access to the corresponding account and automatically redirecting the user to the Import page. If the code is 401, an unauthorized client error message is displayed. The Login page also checks for empty token submissions and employs dynamic input validation to ensure that the token is correctly formatted before it is submitted. After successful authentication, the navigation bar item changes to display Logout instead of Login.

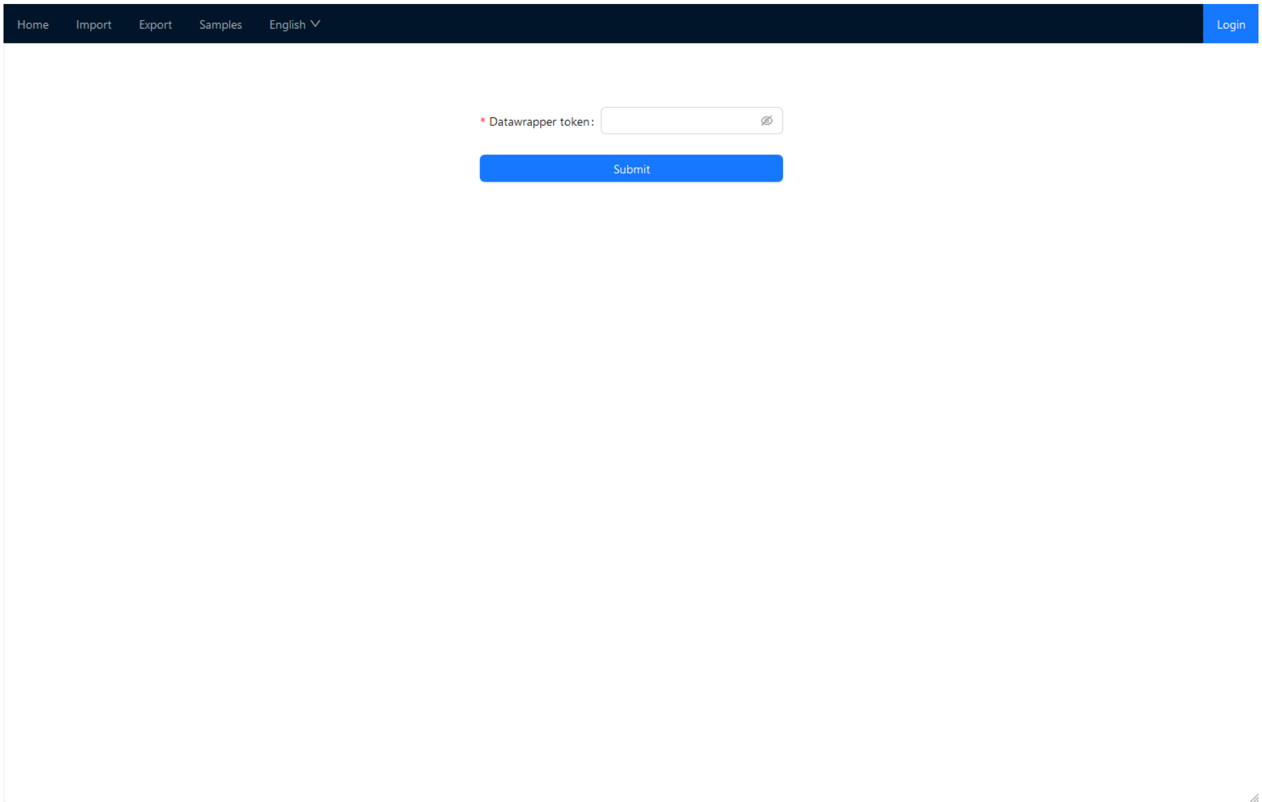

Figure 3.5 Login page

Import and Samples Pages

The Import (Figure 3.6) and the Samples (Figure 3.7) pages are intended for selecting an input chart. They share a similar layout and are designed to display charts in the database in descending order by creation date, placing the most recently created charts on the first page. Each chart is represented by a thumbnail, and clicking on any thumbnail activates a preview on the right side of the page. This ensures that users can quickly and easily access the specific data visualization they need.

Import page. The Import page is protected by a private route, requiring user authentication for access. Any unauthorized user attempting to visit the page is redirected to the Login page. Once authenticated, the Import page fetches the list of visualizations from the user's account using the `charts` API with `limit` and `offset` query parameters set according to the number of visualizations per page and the current page number. Due to the project's defined scope, the button in the bottom right corner for selecting input is only activated when the selected visualization is within the range of supported chart types.

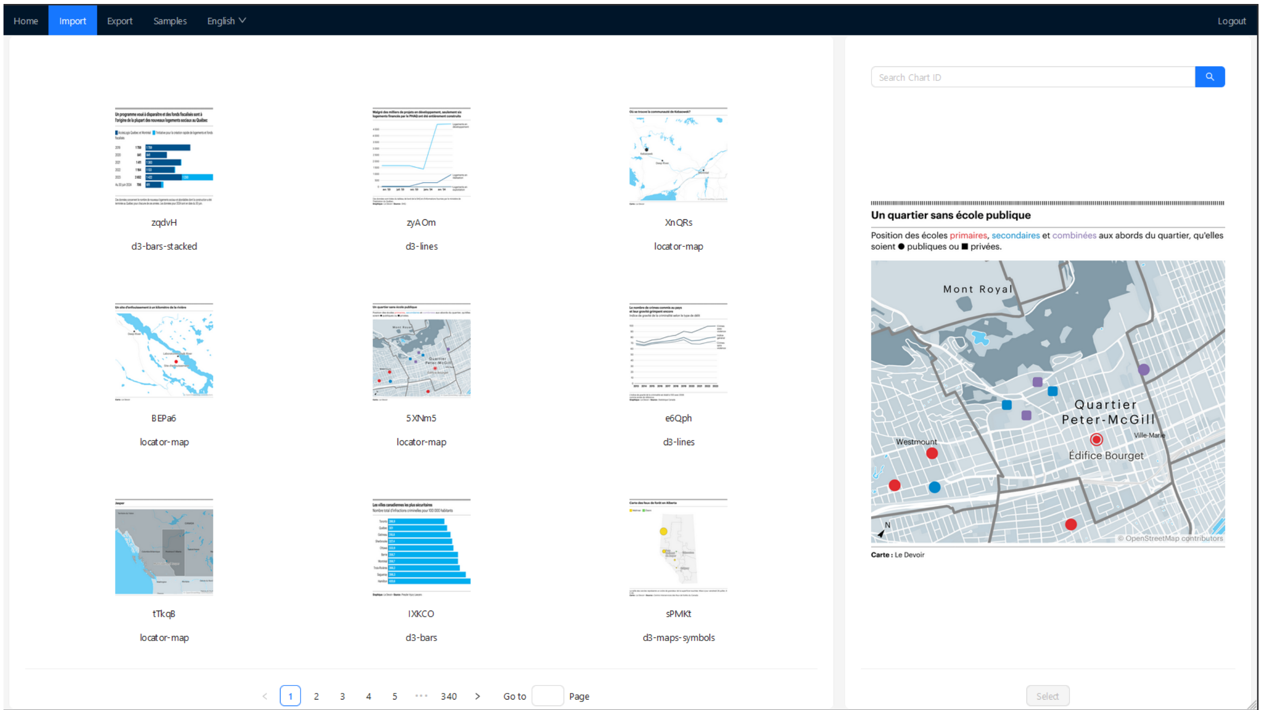

Figure 3.6 Import page

Samples page. The Samples page contains examples from predefined datasets, which are subsets of charts from the accounts of Le Devoir and Radio Canada. When a user clicks on the page's navigation bar item, a dropdown menu appears, offering links to the corresponding datasets. The samples have been selected to include only those created between 2021 and 2024 that align with the supported types, allowing the research prototype's capabilities to be showcased without requiring the user to log in. Certain charts are excluded from the datasets due to specific characteristics that render them incompatible. The criteria for determining this compatibility will be discussed in the backend section.

Home page

After selecting an input chart, the user is automatically directed to the Home page (Figure 3.1), the primary interface where a list of detected features, a section for generated descriptions, and the input chart are displayed. Users can choose features, edit descriptions, and reorder them by dragging the tags. The descriptions are visually linked to the input chart using hover-activated annotation techniques, which are tailored to the specific chart type and feature. Further details on these techniques are available in Table A.1 in the appendix. In the table, "None" indicates the absence of any annotation, while "N/A" means that the feature is not applicable to the specific chart type.

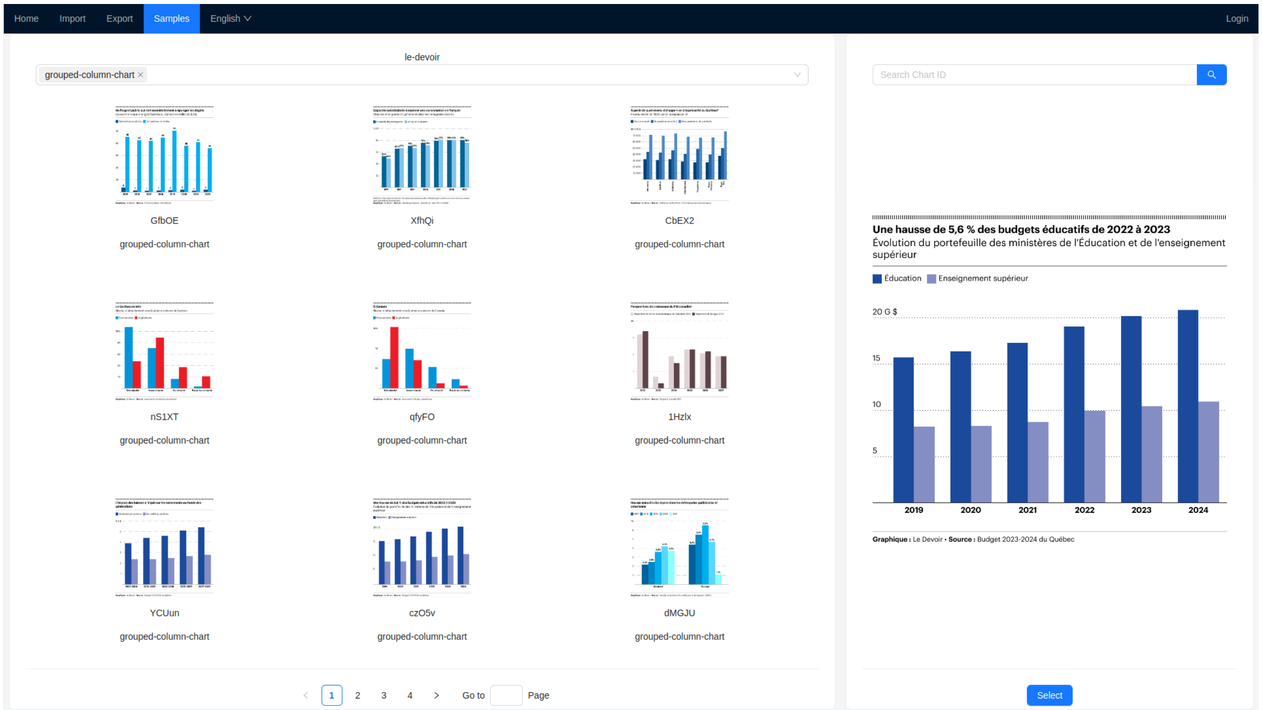

Figure 3.7 Samples page

Export page

The Export page (Figure 3.8) is designed to organize the descriptions created on the Home page into coherent paragraphs. The process involves concatenating individual description segments to form narratives. Below each paragraph, users have the option to further refine the content by clicking the edit button, allowing them to modify sentences as needed. Additionally, a copy button is placed next to the edit button, enabling users to easily copy the text for use in other documents or applications.

3.4.3 Backend

The backend is responsible for implementing REST APIs that facilitate communication between the Datawrapper database and the frontend. These APIs are designed to fetch and process data, ensuring that the information is prepared in a format suitable for frontend use. To achieve this, the backend employs Python's scientific computing libraries, such as NumPy and Pandas, which provide powerful tools for data analysis and numerical computation. Additionally, the backend is responsible for generating language content based on predefined templates. This language generation step involves dynamically crafting text output according to a set of specific conditions.

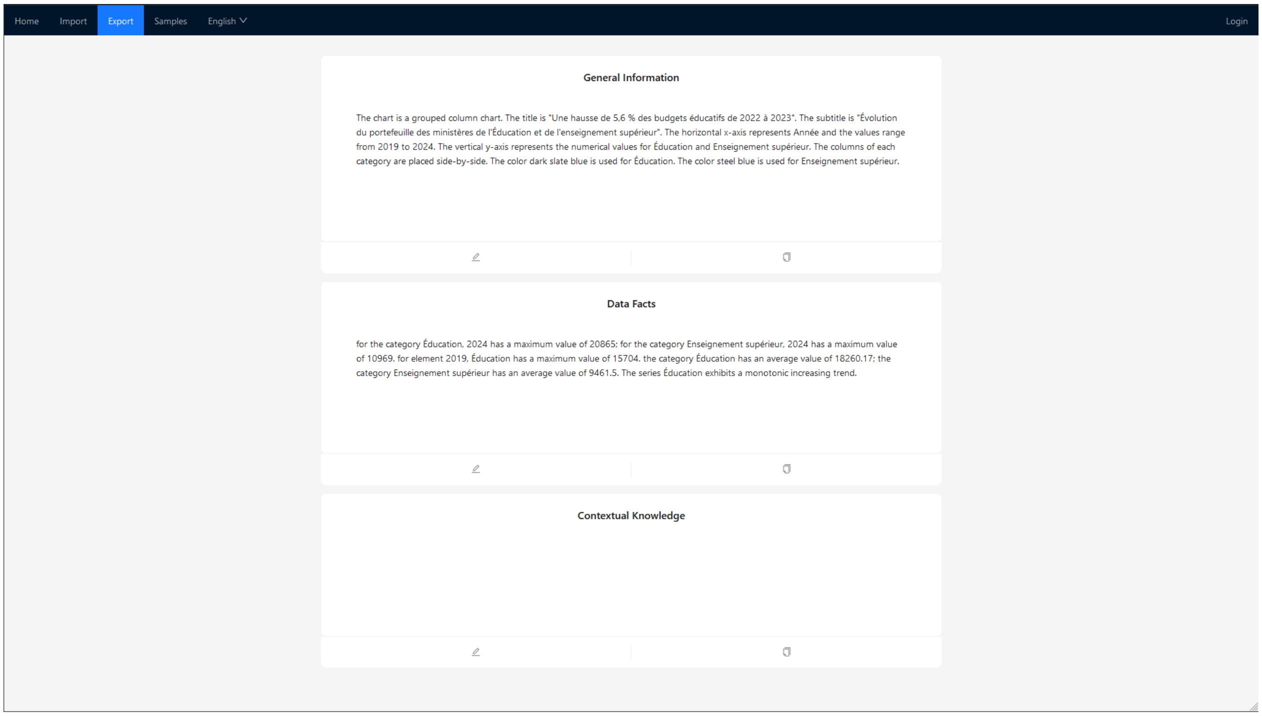

Figure 3.8 Export page

Chart Data Extraction and Preprocessing

Once a user selects an input chart, the backend fetches the necessary data using Datawrapper APIs. To effectively handle the responses, it is important to first understand the process of creating a visualization using Datawrapper, which consists of four key steps:

1. **Uploading Data**: Users can upload raw data by copying and pasting a data table, uploading an XLS or CSV file, connecting to a Google Sheet, or linking an external data source.

2. **Checking and Describing**: Datawrapper processes the uploaded data by converting it into a table and automatically identifying the data types. Users may need to make adjustments, correct any errors, and ensure the data is accurately represented, such as adding units of measurement.

3. **Visualizing**: Users choose a visualization type, refine the axis labels and the encoding colors, and add a title, subtitle, footnote, or other annotations to enhance the clarity and presentation of the visualization.

4. **Publishing and Embedding**: Users can export the final visualization by generating

an interactive inline frame or a static image, which can then be embedded into websites or shared across platforms.

Information about these four steps can be retrieved from the database using the chart ID and the token associated with the user's account. Table 3.2 shows the types of information required by the framework and the APIs associated with the GET requests. The following section will explain the structure of the response returned by each API and the preprocessing step.

Table 3.2 Extracted data

Type of information	API
Metadata in JSON format	`https://api.datawrapper.de/v3/charts/id`
Raw data	`https://api.datawrapper.de/v3/charts/id/data`
Chart in SVG format	`https://api.datawrapper.de/v3/charts/id/export/svg`

Metadata in JSON format. The metadata request returns information and configuration settings for a visualization. The top-level properties include key attributes such as the chart ID, type, title, publication date, and a metadata property, which contains personalized settings applied by the user after uploading the raw data. The code block below illustrates a partial response from the request for the chart displayed in Figure 3.1.

```
{
  "id": "cz05v",
  "type": "grouped-column-chart",
  "title": "Une hausse de 5,6 % des budgets éducatifs de 2022 à 2023",
  "publishedAt": "2023-03-22T01:55:16.000Z",
  ...
  "metadata": {
    "data": {
      "transpose": true,
      "horizontal-header": true,
      "changes": [{...}, {...}, ...],
      ...
    },
    "describe": {
      "intro": "Évolution du portefeuille des ministères de l'Éducation...",
      ...
```

```
    },
    "annotate": {
      "notes": ""
    },
    "visualize": {...},
    ...
  }
}
```

The metadata property contains objects such as `data`, `describe`, `annotate`, and `visualize`. The `data` object records the modifications the user made to the raw data during step 2 “Checking and Describing”, including whether the data table has been transposed and if it has a horizontal header. Additionally, it logs all the changes made to the data table in chronological order, specifying the specific row and column numbers affected, timestamps, previous values, and the updated values for each adjustment. The `describe` and the `annotate` objects store the user’s inputs for annotating the chart during step 3 “Visualizing”, like the optional introduction below the title and the notes at the bottom of the chart. The `visualize` object, which varies based on the type of visualization, also includes modifications made during step 3, such as whether the data is sorted and the chosen sorting order.

Raw data. The request for raw data returns the input from step 1 “Uploading Data” in string format. The response is then preprocessed according to the chart type and its metadata to reconstruct the adjusted data used for the actual visualization. The preprocessing steps begin with converting the raw data into a Pandas DataFrame for more efficient manipulation and analysis. This conversion process is automated using CSV Sniffer, a set of functions that heuristically detect the delimiter character in use. Then, for each chart type, the DataFrame is modified based on whether it has a horizontal header, ensuring that the structure aligns with the intended format. If the data was transposed in step 2 “Checking and Describing”, the DataFrame is transformed accordingly. Next, the framework applies recorded changes by rewriting each modification to the corresponding cell in the DataFrame. To ensure accuracy, it compares the previous values against the raw or modified data in its current state. This guarantees that the final data reflects the preprocessed steps taken during the initial visualization creation process in Datawrapper.

Given the nature of the datasets, the data values are typically in string format rather than numerical. Moreover, most of the datasets originate from French sources, where decimals use commas instead of periods. Some data tables also include units of measurement, such as percentage signs or currency symbols. To address this, the framework first removes the units of measurement and identifies whether the values are in English or French. It then converts these values from strings to numbers based on the identified conditions and sorts the DataFrame according to the specifications provided by the `visualize` object. In some cases, errors may occur during these preprocessing steps. For example, during the change recovering phase, the actual data might not align with the previous changes tracked in the database. When this happens, the backend responds by throwing an error, while the frontend displays the corresponding error message to the user, highlighting the inconsistency.

Chart in SVG format. The request for a chart in SVG format returns the vector image of the chart. This feature is available exclusively for users with a valid token generated under a paid account. Upon retrieval, the SVG file is parsed using Beautiful Soup, a Python package for parsing HTML and Extensible Markup Language (XML) documents. This process is complemented by the use of regular expressions to extract the RGB color codes based on predefined IDs associated with each chart type. Once the relevant color codes are extracted, they are employed to reconstruct the visual elements of the chart. Subsequently, these RGB color values are translated into their corresponding color names in both English and French. This bilingual color naming is then utilized in generating the descriptive text output.

Language Generation

Information in the extracted and preprocessed data is categorized based on its semantics to facilitate the creation of natural language content. The process involves combining predefined textual structures with variables, using templates as blueprints. These templates contain fixed phrases or sentences with placeholders that are filled with specific text or numbers during runtime. This approach allows for customized natural language generation by dynamically inserting relevant information, making it useful in scenarios requiring consistent output. Information about the input chart is systematically classified into general information and data facts. Templates are then applied depending on the chart type and its specific features, ensuring the generated text accurately reflects the unique characteristics of each chart.

General Information. General information describes the overall structure and encoding channels of the input chart. Elements like the chart type, title, subtitle (intro), and footnote (notes) are extracted from the metadata. Information about the axes is derived from the DataFrame's header and the range of its values. Colors are extracted from the SVG. To handle color identification, we used the list of CSS3 extended color names which consist of key-value pairs of RGB hex color codes and their corresponding names. The RGB color codes are converted to CIELAB color space [88], as it aims to approximate human vision rather than just mimic the physical attributes of colors, making it more suitable for tasks requiring perceptual accuracy, such as color matching. We stored the array of CIELAB codes along with an array of dictionaries containing both English and French color names in a JSON file, maintaining the same order. Unfortunately, there is no official French version of the CSS3 color names, so we sourced some translations from the internet and manually translated a few others. For any input chart, the framework first converts the RGB color codes to CIELAB and then applies the nearest neighbor algorithm [89] to determine the index of the closest color match. This index is subsequently used to retrieve the corresponding English and French color names. This process ensures that the closest color name can be effectively identified for each instance of detected color in the input chart.

Each chart type has a template available in both English and French. Figure 3.9 provides an example of the template for the general information section of a bar chart in English. Brackets with ellipses indicate placeholders that will be populated with specific text and numbers extracted from the input.

```
d3-bars
└─ general information
   ├─ type: The chart is a [bar chart]
   ├─ title: The title is [...]
   ├─ intro: The subtitle is [...]
   ├─ notes: The footnote is [...]
   ├─ axes
   │  ├─ x-axis: The horizontal x-axis represents [...] and the values
   │  │  range from [...] to [...]
   │  └─ y-axis
   │     ├─ numerical: The vertical y-axis represents [...] and the
   │     │  values range from [...] to [...]
   │     └─ categorical: The vertical y-axis represents [...] and
   │        contains the following items: [...]
   └─ color
      ├─ single-color: The bars are plotted in [...]
      └─ multi-color: The color [...] is used for [...]
```

Figure 3.9 Template for the general information section of a bar chart

Data Facts. Data facts include data points and statistical measures calculated from the preprocessed DataFrame, such as extrema, mean, standard deviation, and median. To determine how these values are computed for each chart, we need to first take a look at Table 3.3, which shows the data structures that each chart type supports, where C stands for a categorical variable and N stands for a numerical variable. Consider the example of a data frame in Table 3.4, where column X is the independent variable, and columns Y_1 and Y_2 are the dependent variables. In this case, the data structure is denoted as C * N * N if X is categorical, or N * N * N if X is numerical.

The template used for generating text output depends on the chart type and the data structures that it accommodates. When statistical values are computed for columns Y_1 and Y_2, we refer to this as category-wise computation. Conversely, when values are computed across rows, it is termed element-wise computation. For area and line charts with multiple dependent variables, the framework enables category-wise calculation of statistical values. For grouped bar charts and grouped column charts, values can be analyzed both by category and by individual items, identified by indices. Stacked charts have an additional option of aggregated values alongside values specific to each category and element. In terms of outliers, we employed the Interquartile Range [90] method. This statistical technique is designed to identify anomalies by defining acceptable boundaries based on the spread of the data. If the chart involves a numerical independent variable, the framework also computes correlation and trend. To analyze numerical data exhibiting a trend, our initial step is to determine whether the data exhibits a monotonic behavior. If the data are not monotonic, we proceed by segmenting the data into intervals where the values are either rising, falling, or remaining constant. Should the total number of these intervals exceed a specific threshold, it indicates considerable variability in the data's trend. In such cases, we further refine our analysis by calculating the slope for each interval. This calculation allows us to identify and highlight those intervals that demonstrate significant changes with respect to the independent axis, thereby focusing on the most critical variations within the data.

Figure 3.10 and Figure 3.11 illustrate templates for the data facts section of a bar chart and a grouped bar chart in English. A trailing semicolon at the end of a sentence indicates that it can be concatenated with other phrases to form a longer, more complex sentence. For example, in a time-series chart with multiple increases along the x-axis, the template may read "the data points increase from [...] to [...] and [...] to [...]". For a grouped bar chart presenting multivariate data, statistical values can be calculated within individual groups or across multiple groups. This allows for a more detailed analysis, enabling comparisons between different data series. The template is designed to accommodate the complexity of data with multiple variables, providing a structured way to convey insights derived from

Table 3.3 Data structures supported by each chart type

Chart type	Univariate		Multivariate	
	C * N	N * N	C * N * ... * N	N * N * ... * N
d3-bars	✓	✓		
d3-bars-split			✓	
d3-bars-stacked			✓	✓
d3-bars-grouped			✓	✓
column-chart	✓	✓		
grouped-column-chart			✓	✓
stacked-column-chart			✓	✓
d3-area		✓		✓
d3-lines		✓		✓
d3-pies	✓			

Table 3.4 Example of a data frame

Index	X	Y_1	Y_2
0	x_0	$y_{0,1}$	$y_{0,2}$
1	x_1	$y_{1,1}$	$y_{1,2}$
2	x_2	$y_{2,1}$	$y_{2,2}$
3	x_3	$y_{3,1}$	$y_{3,2}$

category-specific and element-specific statistical computations.

```
d3-bars
  data facts
    data-points:  The chart has the following data:  [...]
    extrema
      maximum:  [...]  has a maximum value of [...]
      minimum:  [...]  has a minimum value of [...]
    distribution
      mean:  The average value is [...]
      std:  The standard deviation is [...]
      median:  The median value is [...]
    outliers:  [...]  has a potential outlier of [...];
    correlation
      positive:  The data shows a [strong/weak] positive correlation
      no-correlation:  There is no correlation observed in the data
      negative:  The data shows a [strong/weak] negative correlation
    trend
      monotonic:  The data exhibits a [increasing/constant/decreasing]
      trend
      non-monotonic:  The data points [increase/decrease/remain
      constant] from [...]  to [...];
```

Figure 3.10 Template for the data facts section of a bar chart

```
d3-bars-grouped
└─data facts
  ├─data-points: The category [...] has the following data: [...];
  ├─extrema
  │ ├─maximum
  │ │ ├─category-wise: For the category [...], [...] has a maximum
  │ │ │ value of [...];
  │ │ └─element-wise: For element [...], [...] has a maximum value of
  │ │   [...];
  │ └─minimum
  │   ├─category-wise: For the category [...], [...] has a minimum
  │   │ value of [...];
  │   └─element-wise: For the element [...], [...] has a minimum
  │     value of [...];
  ├─distribution
  │ ├─mean
  │ │ ├─category-wise: The category [...] has an average value of
  │ │ │ [...];
  │ │ └─element-wise: The element [...] has an average value of
  │ │   [...];
  │ ├─std
  │ │ ├─category-wise: The category [...] has a standard deviation of
  │ │ │ [...];
  │ │ └─element-wise: The element [...] has a standard deviation of
  │ │   [...];
  │ └─median
  │   ├─category-wise: The category [...] has a median value of
  │   │ [...];
  │   └─element-wise: The element [...] has a median value of [...];
  ├─outliers: For the the category [...], [...] has a potential
  │ outlier of [...];
  ├─correlation
  │ ├─positive: The series [...] shows a [strong/weak] positive
  │ │ correlation
  │ ├─no-correlation: There is no correlation observed in the series
  │ │ [...]
  │ └─negative: The series [...] shows a [strong/weak] negative
  │   correlation
  └─trend
    ├─monotonic: The series [...] exhibits a [increasing/constant/decreasing]
    │ trend
    └─non-monotonic: The data points for [...]
      [increase/decrease/remain constant] from [...] to [...];
```

Figure 3.11 Template for the data facts section of a grouped bar chart

CHAPTER 4 EVALUATION

This chapter presents a comprehensive evaluation of the framework. We will begin by exploring three usage scenarios that illustrate the application of the prototype in real-world contexts, showcasing its adaptability to various input types. These scenarios are carefully selected to illustrate the framework's flexibility in handling different data structures and user interactions, highlighting its versatility. Following this, we will evaluate the user experience by analyzing the results of a user study conducted with three participants, focusing on how the framework meets the established design requirements. This evaluation not only validates its design but also guides future development and works.

4.1 Usage Scenarios

A user who created a chart with Datawrapper and wishes to use the framework must first obtain a token associated with their account. This can be done by navigating to the settings in Datawrapper and generating a token, as illustrated in Figure 4.1. Afterward, the user needs to store this token securely and log into the framework using it. If the chart was recently created, it will automatically appear as the first visualization listed on the Import page. Otherwise, the user can locate it by copying its ID from Datawrapper and pasting it into the search bar to select the chart.

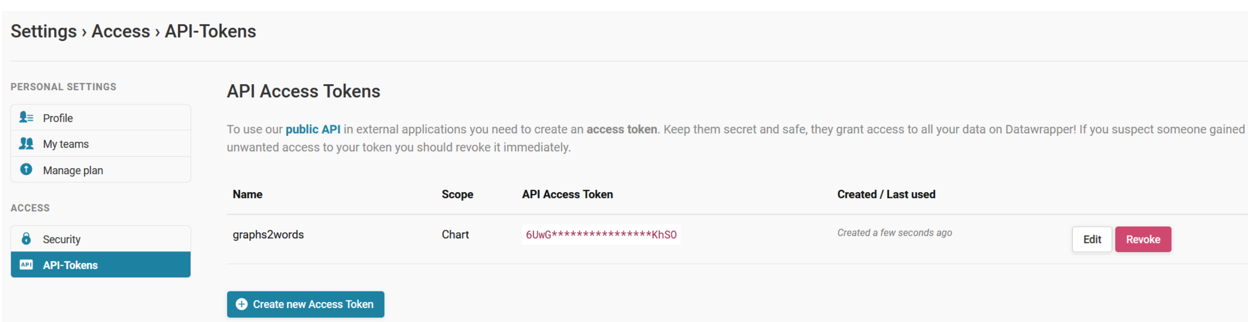

Figure 4.1 API access tokens in Datawrapper

The framework supports a wide range of chart types. To effectively describe usage scenarios, we will categorize these types based on their underlying data characteristics. Specifically, we distinguish between charts with univariate or multivariate data, and those with categorical or numerical independent variables. In the following section, we will explore three different cases.

4.1.1 Univariate Categorical Data

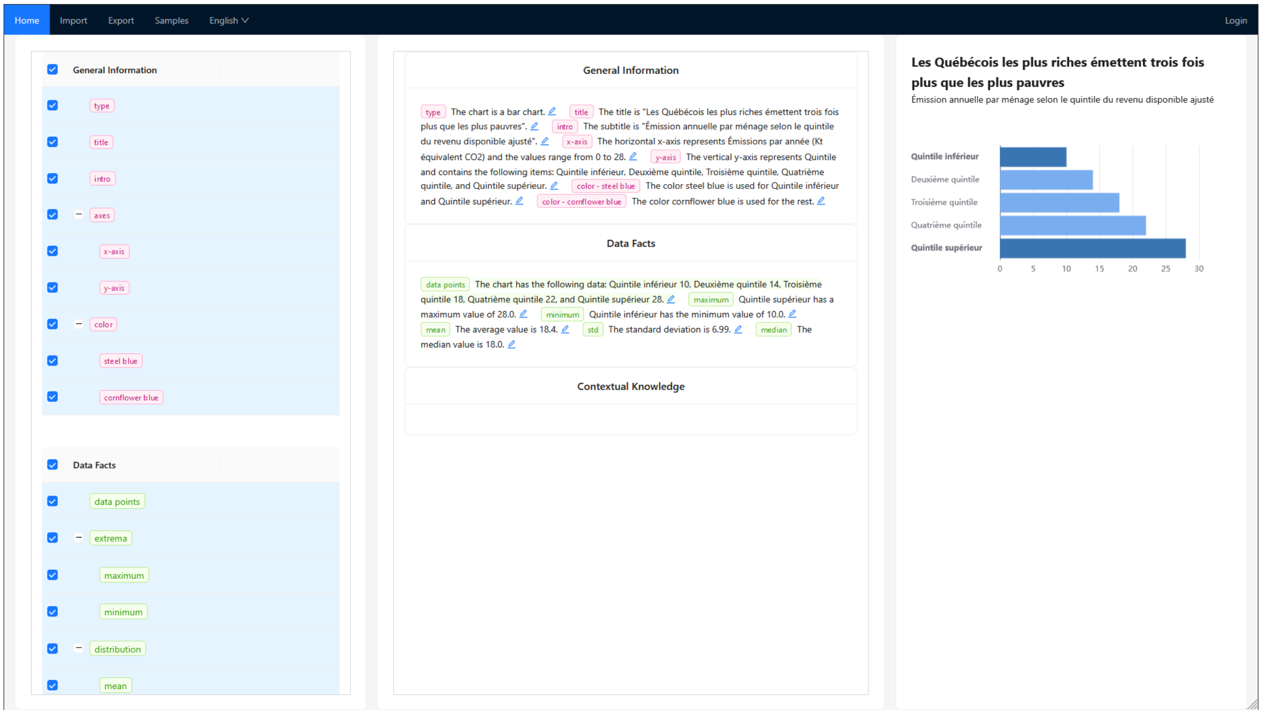

Figure 4.2 Simple bar chart

For a simple bar chart, as shown in Figure 4.2, the framework's main interface displays a list of chart features on the left and a description section in the center. Both sections are organized into three parts: general information, data facts, and contextual knowledge. Each feature is linked to a description. Users can select checkboxes to view the generated texts for specific features and hover over them to trigger an animated visual cue on the chart, highlighting the corresponding area to illustrate their meaning. Additionally, users can edit and reorder the descriptions. Once satisfied, they can click on the export button in the navigation bar to copy and paste the full descriptions.

Without any editing, the generated text for the general information section is as follows: "*The chart is a bar chart. The title is "Les Québécois les plus riches émettent trois fois plus que les plus pauvres". The subtitle is "Émission annuelle par ménage selon le quintile du revenu disponible ajusté". The horizontal x-axis represents Émissions par année (Kt équivalent CO2) and the values range from 0 to 28. The vertical y-axis represents Quintile and contains the following items: Quintile inférieur, Deuxième quintile, Troisième quintile, Quatrième quintile, and Quintile supérieur. The color steel blue is used for Quintile inférieur and Quintile supérieur. The color cornflower blue is used for the rest.*" And the description for the data facts section is: "*The chart has the following data: Quintile inférieur 10, Deuxième* *quintile 14, Troisième quintile 18, Quatrième quintile 22, and Quintile supérieur 28. Quintile supérieur has a maximum value of 28.0. Quintile inférieur has the minimum value of 10.0. The average value is 18.4. The standard deviation is 6.99. The median value is 18.0.*" The descriptions provide a clear overview of the chart's structure and data, laying the foundation for users to understand its key elements and insights. With this baseline, users can further refine and customize the descriptions to better suit their specific needs or audience.

4.1.2 Multivariate Categorical Data

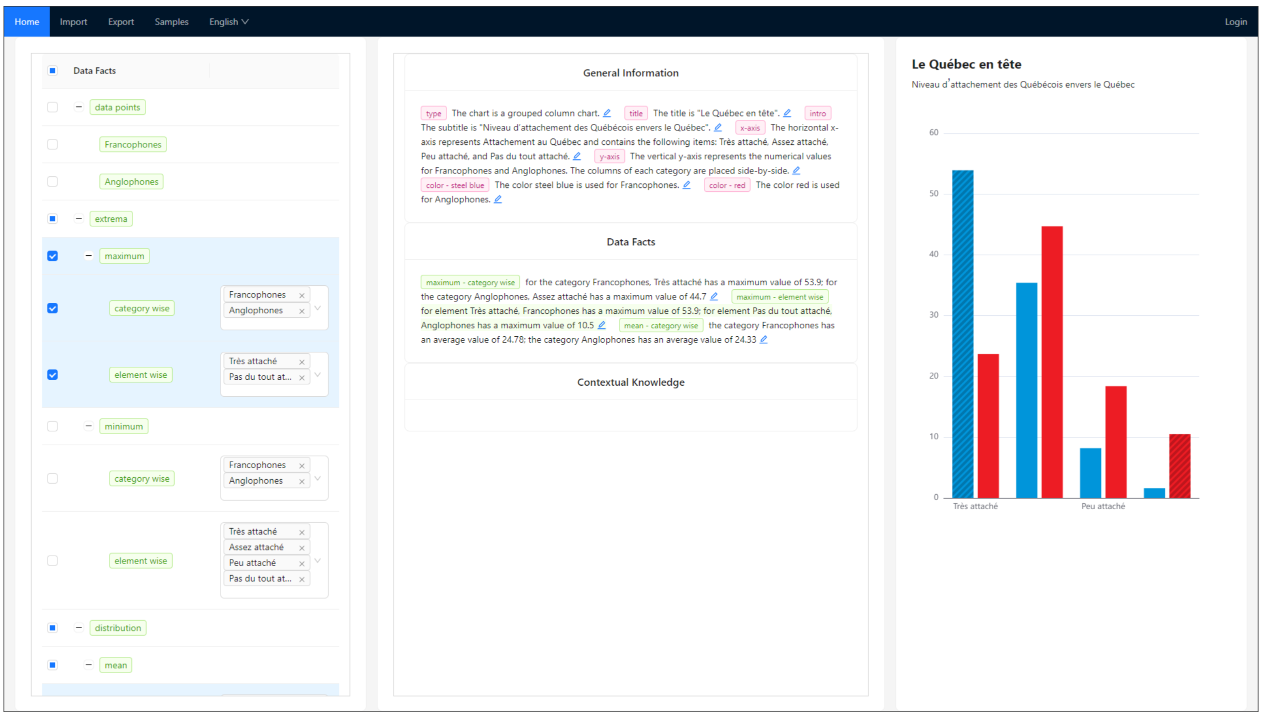

Figure 4.3 Grouped column chart

Multivariate data refers to data involving multiple variables, typically more than one dependent variable, which are analyzed simultaneously to understand their relationships or to compare them across different categories. Consider the chart in Figure 4.3 that visualizes the levels of attachment to Quebec among two groups of people: French speakers and English speakers. The chart is organized into four distinct sets of columns over the x-axis, each corresponding to a specific level of attachment. From left to right, the sets of columns are: very attached, quite attached, a little attached, and not attached. Within each set, two columns are displayed side by side—one in blue representing French speakers and the other in red for English speakers.

The section for general information functions similarly to the one for the bar chart, where selecting each feature generates a corresponding segment of the description. The generated description reads: *"The chart is a grouped column chart. The title is "Le Québec en tête". The subtitle is "Niveau d'attachement des Québécois envers le Québec". The horizontal x-axis represents Attachement au Québec and contains the following items: Très attaché, Assez attaché, Peu attaché, and Pas du tout attaché. The vertical y-axis represents the numerical values for Francophones and Anglophones. The columns of each category are placed side-by-side. The color steel blue is used for Francophones. The color red is used for Anglophones."* However, the structure of the feature list in the data fact section differs. In a simple bar chart with a single dependent variable, identifying the maximum value is straightforward, as there is only one peak value. In contrast, a grouped column chart like this requires a more nuanced analysis. For example, one might seek to determine which attachment level has the highest value specifically for French speakers or compare which group of people has a higher value at a particular attachment level. Consequently, the maximum value can be analyzed either by category (e.g., by language group) or by element (e.g., by attachment level).

In such cases, an additional dropdown column is provided in the feature list, allowing users to select specific variables. By default, the dropdowns contain all variables, and users can choose to clear the selections and add or remove variables as needed. For instance, when determining the maximum value by category, the user might select both variables for French speakers (Francophones) and English speakers (Anglophones). The corresponding generated sentence would be: *"for the category Francophones, Très attaché has a maximum value of 53.9; for the category Anglophones, Assez attaché has a maximum value of 44.7."* The same principle applies when analyzing the maximum value element-wise. If the user is only interested in the values for very attached (Très attaché) and not attached (Pas du tout attaché), the generated text would read: *"for element Très attaché, Francophones has a maximum value of 53.9; for element Pas du tout attaché, Anglophones has a maximum value of 10.5."* When hovering over this text, as shown in the figure, the corresponding bars are highlighted with a dashed texture, visually emphasizing the referenced observations. This approach allows for flexible and detailed analysis, enabling users to compare values within and across categories with visual cues that reinforce the connection between the generated text and its representation.

4.1.3 Multivariate Numerical Data

The cases discussed above involve categorical independent variables. Figure 4.4, however, presents a line chart depicting time-series data, which differs in nature. The chart features two dependent variables: one representing the number of new COVID-19 cases per million

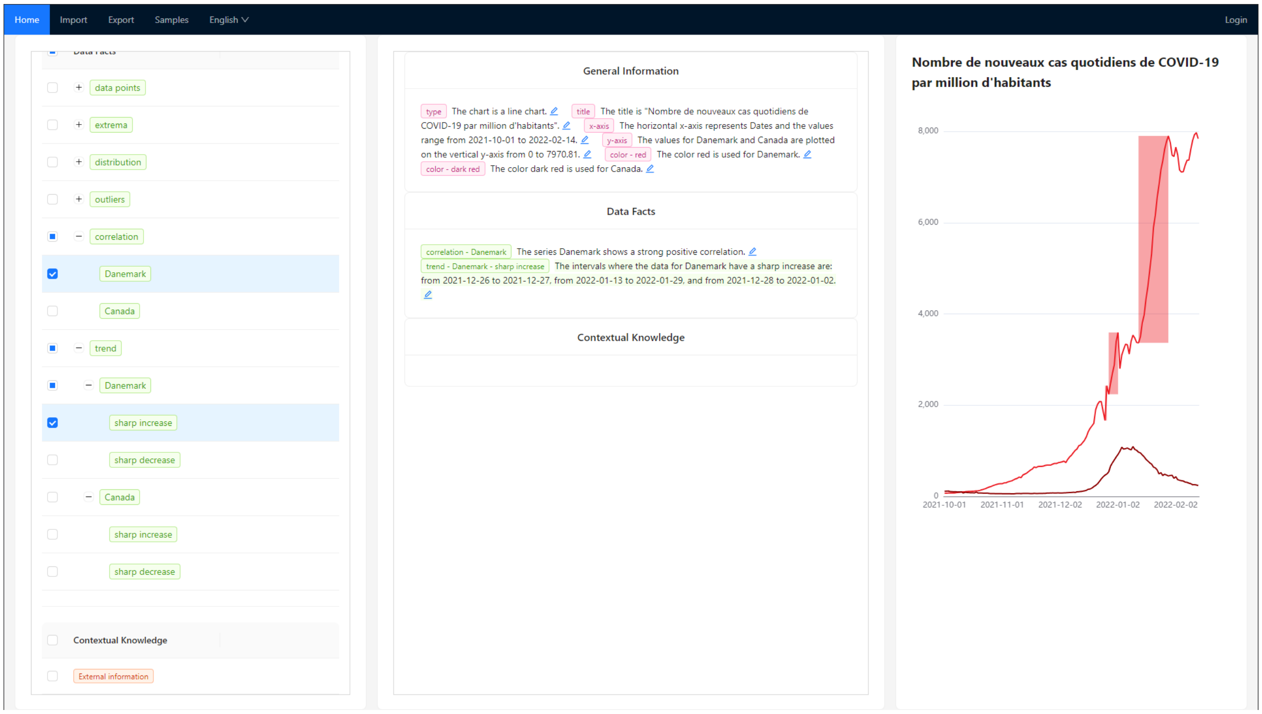

Figure 4.4 Line chart

inhabitants in Denmark, and the other for Canada. The data consists of observations recorded at specific time intervals, where the sequence is crucial as it reflects changes over time.

Given the fact that the chart's independent variable is numerical and continuous, additional features are provided beyond variable selection via dropdowns. These include options to analyze correlation and trends within each data series. As illustrated in the figure, the framework identifies a strong positive correlation in the Denmark series and pinpoints intervals of significant data increases with the generated text: *"The intervals where the data for Danemark have a sharp increase are: from 2021-12-26 to 2021-12-27, from 2022-01-13 to 2021-01-29, and from 2021-12-28 to 2022-01-02."* When users hover over this text, the corresponding intervals are dynamically highlighted on the chart.

The framework not only allows users to explore categorical variables but also provides advanced functions for analyzing time-series data. By integrating features like correlation and trend detection, the framework enhances the user's ability to interpret complex data patterns over time, making the insights actionable.

4.2 User Study

We conducted a user study[1] with three participants (**P1-P3**), all of whom specialize in media content creation. The participants are experienced journalists, with an average of 10 years of professional experience and prior experience using Datawrapper for creating data visualizations. Each participant was presented with the prototype and given approximately one month to explore its features on their own. During this period, they had the option to experiment with charts from the provided samples or to create and import a new chart of their choice. The samples page includes two datasets: one from Le Devoir comprising 401 charts and another from Radio Canada comprising 1035 charts. These datasets, which consist of charts created over the last three years, offer a variety of chart types based on tabular data, providing participants with a rich set of examples to explore.

Following this exploratory phase, we conducted individual semi-structured interviews, each lasting approximately 30 minutes, to collect feedback on the user experience. The interviews featured both qualitative and quantitative components. For the qualitative study, we posed a series of open-ended questions to assess whether our design met the established requirements. For the quantitative study, participants were asked to complete an online survey individually at the end of the interview. The survey included single-choice statements rated on a 7-point Likert scale, along with a few demographic questions that did not include any identity-related inquiries. As a token of appreciation, participants who completed both the interview and the survey received a 10$ Amazon eGift Card.

4.2.1 Qualitative Study

The qualitative study draws inspiration from the thinking aloud test used in user-centered studies and the heuristic evaluation method found in expert-driven studies. In user-centered studies, the thinking aloud test involves participants verbalizing their thoughts, feelings, and decision-making processes as they interact with an interface. This approach provides real-time insights into the user's cognitive processes, effectively capturing their experiences and verifying design decisions based on direct user feedback. On the other hand, heuristic evaluation is a method in which experts assess the interface against established usability principles, known as heuristics. While the thinking aloud test focuses on the end-user perspective, heuristic evaluation relies on the expertise of professionals to identify potential usability issues based on best practices and standards. It is efficient in detecting major design flaws instead of context-specific issues that users might encounter.

[1] User study approved by Research Ethics Board of Polytechnique Montréal (ID CER-2324-51-D)

In our case, participants are considered both end-users and industry-specific experts, bringing a deep understanding of usability issues and domain-specific requirements that are crucial for evaluating specialized interfaces. By combining the strengths of both approaches, the study aims to achieve a comprehensive understanding of user experience, balancing user needs with expert insights. We had three participants for this study, a number that aligns with the recommended range of three to five evaluators for both the thinking aloud method and heuristic evaluation [91, 92]. The sample size is deemed sufficient to uncover key issues while maintaining a manageable scope for detailed analysis.

The interview questions, detailed in the appendix, are organized into distinct sections to assess both the usefulness and usability of the framework. The usefulness section explores participants' perceptions of how well the framework facilitates the recognition and communication of key insights within an input chart. We examined users' views on the design, with a focus on features like annotations and their role in aiding the interpretation of findings. We also asked whether the framework supports the creation of comprehensive chart descriptions and evaluated the effectiveness of specific features, including chart selection, the list of important elements, and the ability to edit and reorder the generated descriptions. Additionally, we investigated whether it contributed to saving time and improving efficiency in the process of writing chart descriptions. On the usability side, the questions center on the overall user experience, examining whether participants found the framework easy to use and learn, and if they encountered any difficulties during their interactions. We were also interested in understanding whether its design and functionality were accessible not only to chart creators but also to a broader, less specialized audience. Finally, the follow-up questions provided an opportunity for participants to share additional thoughts from a journalism perspective, including suggestions for missing features or functionalities that could enhance the framework's utility. These questions aimed to gather detailed feedback to guide future iterations, ensuring it meets the needs of its users.

In the following section, we will delve into the user feedback collected during the study, with a focus on how well it aligns with our established design requirements. By examining participants' responses, we aim to assess the effectiveness of the framework in meeting the intended goals. This analysis will provide insights into the strengths of the design, highlight areas for improvement, and offer a deeper understanding of how the framework performs in real-world scenarios.

R1. The design should be compatible with the existing workflow to ensure high usability and a smooth adoption process. Feedback from participants emphasizes the design's ease of use, suggesting that it aligns well with the existing workflow and does not introduce unnecessary complexity. In general, users value the effectiveness and simplicity of the main interface. For instance, **P2** remarked: *"Overall, I really enjoyed using the tool. I think it is very easy to use and understand. Selecting a chart is easy, and I like the fact that when you click on something, it displays immediately. I also like that we can modify the descriptions to add precision. I think it can save a lot of time when it comes time to write an alternative description."* Similarly, **P3** noted: *"The tool is simple and intuitive, in the sense that you can click on what you need without having to search around. I didn't feel that I had to click everywhere to try to understand how to use it. Usually, you click and see, and it works. I didn't feel like I was wasting time or struggling to figure it out. I found it very easy to use."* The feedback indicates that the design supports users' existing processes without disrupting them. The ability to make selections and have quick responses contributes to a user-friendly experience, aligning well with the goal of maintaining compatibility with current practices.

R2. The framework should extract features from the input visualization and present them in a structured format to facilitate a clear understanding of the visualization's composition. Participants had mixed reactions to the volume of information provided by the framework. While some appreciated the comprehensive features, others felt it potentially overwhelming. However, The design handles this by offering options for filtering out unnecessary details, allowing users to focus on what is most relevant to them. **P2** expressed appreciation for the extensive information, stating: *"Most of the important information in a chart is available within the tool. I liked the fact that the list was exhaustive. There are a lot of elements, and I can choose the most relevant ones."* Similarly, **P3** commented on the feature richness, adding: *"There is pretty much everything you need. There is a lot of information, but on the one hand, it's still easy to hide things you don't need. You have to get into the habit of unchecking certain things so that it's less intense. It's still easy to manage the amount of information."* While the framework offers an extensive set of features for detailed data analysis, it also includes customizable filtering options that allow users to control the volume of information displayed on the interface. This balance ensures that most users can access in-depth insights without being too overwhelmed by excessive data.

R3. Each feature needs to have a corresponding description to ensure that the information is fully accessible and understandable to the user. Participants generally found the descriptions helpful in enhancing their understanding of the input charts. **P2** appreciated that the descriptions were accurate and straightforward, making the information easy to grasp. **P1** valued the ability to identify significant moments in the data, noting:

"Being able to spot significant moments is remarkable, this brings out interesting points and reminds us where the increases and decreases are. For a person who only has these descriptions to understand the graph, it gives a basis." **P3** found the inclusion of statistical values to be useful, stating: *"The fact that it gives the max, min, and average, I find that interesting and very useful because that's often what we try to tell. Otherwise, it is going to be a lot of work for us to rewrite all that."* Regarding interactions with the text, both **P2** and **P3** enjoyed the ability to rearrange the descriptions to create a coherent narrative. This flexibility allowed them to tailor the information in a way that best supported their storytelling needs. The positive feedback on these features suggests that the framework aids users in interpreting and communicating the key insights from the graph, while also providing the flexibility needed to adapt the information to different contexts.

R4. Connecting the generated descriptions to the original visualization through linking and brushing techniques for maintaining a clear and intuitive relationship between the output and the original input. The feedback provided by participants indicates a positive reception towards the use of animated visual cues for linking descriptions to the original chart. **P1** found the animation interesting as it allows users to contextualize and tell the story by emphasizing relevant visual elements. **P2** thought that the annotations significantly enhance clarity by effectively conveying what users aim to explain and describe, explaining: *"I found that the tool helps a lot because everything is clear. What we see on the right and the elements on the left, we can see the correlation between the two."* The feedback highlights the effectiveness of visual cues in making the relationship between the charts and their descriptions more intuitive. By clearly linking data insights with their corresponding descriptions, the framework not only improves user comprehension but also enhances the ability to convey a cohesive narrative, making it easier for users to explain and describe the insights drawn from the chart.

Some participants pointed out that the framework not only helps users determine what to write but also facilitates deeper reinterpretation of graphs, leading to new and potentially more meaningful insights. The framework's ability to highlight key elements helps users uncover patterns and trends that might not be immediately apparent, enhancing their understanding of the data. **P2** highlighted that it often allows for a new understanding. **P3** elaborated on its utility, explaining that the framework is good for accessibility but could also be used for analysis. Users who need to analyze data quickly may find it useful to see specific information without the need to go to a spreadsheet for calculations, as searching for information in a spreadsheet can be more challenging. With the framework, relevant details are readily available visually with just a click, making it simple and efficient. By presenting information visually and intuitively, the framework streamlines the analysis process, making it more accessible even to those who may not have specialized expertise. This visual guidance can significantly aid in the writing process, as it allows users to focus on the narrative and insights rather than getting bogged down by the intricacies of data manipulation.

4.2.2 Quantitative Study

In addition to the open-ended questions, we asked the participants to fill out an online survey individually after the meeting. The survey consists of 7-point Likert scale statements designed to evaluate the usability, efficacy, and utility of the framework. The participant's responses are intended to validate the insights gathered during the interviews. However, it is important to note that Likert scale statements are reliable only with a large sample size. A small sample size reduces statistical power, making it difficult to detect trends, resulting in a high margin of error. As a result, the average scores tend to be less stable and more susceptible to fluctuations, which can lead to misleading conclusions. Furthermore, because the survey does not require participants to explain their responses, there is a risk of misinterpreting their answers. The limited number of participants also increases the risk of sampling bias, as the small group may not adequately reflect the diversity of a larger population. Given these limitations, the results presented in this section are primarily descriptive. The data may lack generalizability, making it difficult to draw meaningful conclusions or make informed decisions based on the findings.

Figure 4.5 shows the survey answers. The Likert scale statements are plotted on the vertical axis with scores ranging from 1 to 7 on the horizontal axis. In this scale, 1 indicates "strongly disagree", 2 "disagree", 3 "somewhat disagree", 4 "neutral", 5 "somewhat agree", 6 "agree", and 7 "strongly agree". For each statement, the range of the scores is represented by a horizontal blue line, with the point within the range indicating the average. The survey is divided into three sections:

1. **Usability**: This section includes questions designed to estimate how easily users can interact with the framework. The questions assess the intuitiveness and user-friendliness of the user interface, covering aspects such as ease of learning and the clarity of the design.

2. **Efficacy**: This section focuses on the framework's effectiveness in enhancing the user's ability to perform specific tasks. It evaluates whether the framework improves task efficiency, helps prevent errors, and contributes to better performance in the user's professional duties.

3. **Utility**: This section explores the practical value of the framework within the user's workflow, evaluating its smooth integration with existing processes, the usefulness of its features, and overall user satisfaction.

The following section will provide a detailed analysis of the scores for each section of the survey. We will examine the usability, efficacy, and utility scores individually, exploring how participants rated each aspect of the framework.

Usability. The scores for usability statements range from 3 to 7, with average values between 5 and 6.67, indicating a significant positive agreement among participants. Overall, most statements received relatively high scores, suggesting that the framework is generally perceived as easy to use. For instance, the statement *"I can easily understand how to use the tool"* received an average score of 6.67, with values close to each other, highlighting strong consensus. Following that, *"I can learn how to use the tool quickly"* and *"I can easily understand the purpose of the checklist on the left"* both scored an average of 6.33. However, some statements, such as *"The user interface is easy to understand"*, *"Interactions with the tool are clear and comprehensible"*, *"The tool's usage is straightforward and manageable"*, and *"Overall, the tool is user-friendly"* showed a wider range of scores. This variation suggests that while some users found the interface intuitive, others may have encountered difficulties. These discrepancies indicate that, although the framework generally meets user expectations, there are inconsistencies in its performance for some users.

Efficacy. The efficacy of the framework is generally well-regarded. Similar to usability, the scores range from 3 to 7, with averages generally in the mid-to-high range, between 4.33 and 6.33, indicating a moderately positive agreement. For example, the statement *"The tool can help me avoid making mistakes when writing about specific values"* received the highest score of 6.33, reflecting strong confidence in this aspect of the framework. Likewise, the statement *"The tool can help me avoid omitting details when writing descriptions"* consistently received a score of 6 from all participants, suggesting a high level of trust in the framework's ability to ensure completeness. In contrast, the statements *"Using the tool is more efficient compared to reading a list of accessibility guidelines for describing charts"* and *"The tool makes the task of writing descriptions easier"* exhibited a broader range of scores with an average of 5, highlighting variability in perceived efficiency. The lowest average score within this section is 4.33 for the statement *"If I have to describe charts I created, the tool can save me time"*. This suggests that while some users find the framework time-saving, others do not share this view, potentially due to differences in their workflows or expectations.

Utility. The utility of the framework received mixed feedback, with scores ranging from 2 to 7 across different statements. The average scores, which fall between 4 and 5.67, suggest that while the framework is generally perceived as useful, there are areas where its utility could be improved. For instance, the statement *"The tool can be integrated into my existing workflow without changing too much my habit"* received scores between 3 to 6 with an average of 4.67, indicating that integration into existing workflows can be somewhat challenging for some users. Similarly, the scores for the statement *"The tool offers helpful guidance in writing descriptions"* had the same range, with an average of 5, reflecting overall moderately positive but varied perceptions of the framework's value. The widest range of scores in this section is seen in the statement *"Essential features or functionalities for writing descriptions are supported by the tool"*, which had an average of 4 with scores between 2 and 7. This indicates that while some participants found the framework fully functional, others felt it lacked essential features.

The survey results indicate that the framework generally performs well in terms of usability, efficacy, and utility, with average scores typically in the mid-to-high range, where a midpoint value of 4 represents neutral agreement. However, with only three participants, the results should be interpreted with caution, as they may not be fully reliable or representative of a broader user base. The range of scores highlights the variability in user experiences. While some participants gave the framework high ratings, others faced challenges with certain aspects of the interface. This mixed feedback suggests that, although the framework is effective for some people, it may require further refinement to better serve a broader audience. Addressing areas with lower scores, such as interface complexity, could enhance overall user satisfaction and encourage wider adoption.

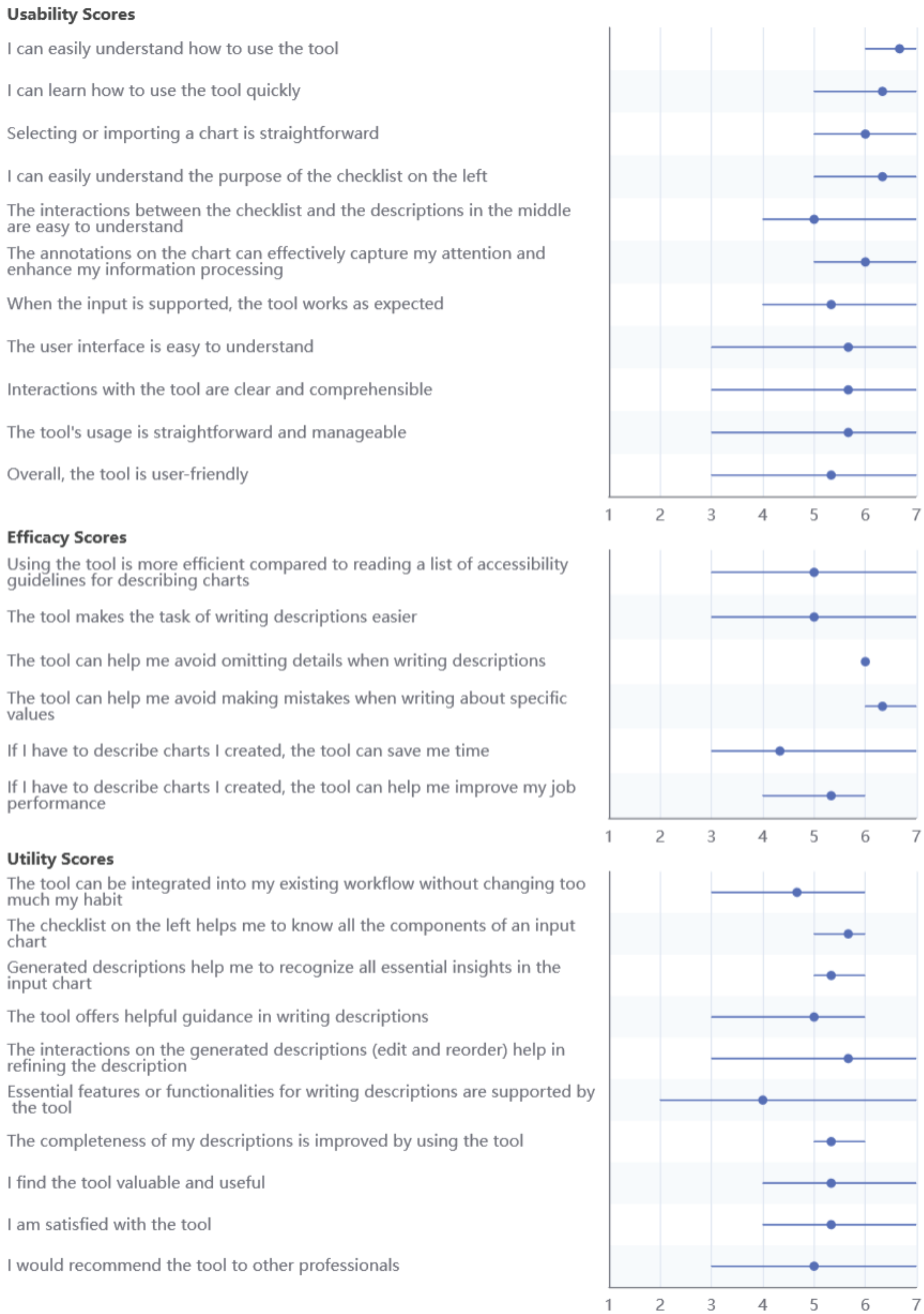

Figure 4.5 Survey scores

CHAPTER 5 CONCLUSION

5.1 Summary of Works

Writing textual descriptions for data visualizations poses considerable challenges, particularly for content creators who lack visual literacy and are unfamiliar with the specific needs of visually impaired people. One of the difficulties lies in determining what to include in these descriptions while ensuring they convey nearly equivalent content. Without a clear understanding of the information needs of visually impaired people for accurate data interpretation, these descriptions can easily become incomplete, leaving out crucial details, or overly simplistic, failing to capture the nuances of the data. This can result in a gap in accessibility, where the content intended to aid visually impaired users falls short of providing the necessary insights. To address these challenges, it is important to equip content creators with the necessary knowledge and tools to produce meaningful and accessible textual descriptions for a diverse audience.

The framework we developed represents a step forward in making data visualizations more accessible, particularly for people with visual impairments. By providing a structured approach for transforming visual content into descriptive narratives, this tool enables content creators to convey complex data insights through text, ensuring that essential information is not lost for those relying on screen readers. The feedback from the evaluation indicates that the framework not only enhances the completeness of the description but also fosters new ways of interpreting and engaging with data. However, further refinement is necessary to fully address the diverse needs of all users. As digital content continues to evolve, integrating accessibility into the core of content creation will be essential, especially considering that a portion of the population prefers text-only content [93]. This project offers a promising foundation for such efforts, ensuring that digital experiences are inclusive and accessible to all users.

5.2 Limitations

While the framework enhances the accessibility of data visualizations through textual descriptions, it is important to acknowledge several limitations. One significant concern is the risk of information overload, where content creators may struggle to strike a balance between providing comprehensive details with keeping descriptions concise for readers, especially in the absence of guidelines from the accessibility department. Additionally, although the framework is designed to improve efficiency in producing textual descriptions, some users felt that the time required to create these narratives outweighs the perceived benefits, particularly when dealing with complex visualizations. Furthermore, the current design of the framework overlooks the explorative aspects of the resulting narrative, as it primarily focuses on translating visual data into text, which leads to a potential for information loss. These limitations suggest areas for further development to enhance the framework's effectiveness and user experience.

5.2.1 High Volume of Information

When describing images for visually impaired individuals, untrained describers may unintentionally omit details, making it difficult for the target audience to fully visualize the content in their minds. To address this, the framework is designed to extract and present all available information within an input chart. While some participants appreciated the comprehensive detail provided, others expressed concerns that the volume of information can potentially overwhelm average users, particularly those with limited familiarity with visual literacy or basic statistical concepts.

The framework includes functionality for users to select and deselect information within an input chart. However, the process still necessitates reviewing the entire list of features, which can be time-consuming. While it is relatively straightforward to filter out unnecessary details, this underscores the challenge of balancing the delivery of comprehensive descriptions with the need to provide a more streamlined, user-friendly experience through an overview or summary. Achieving this balance is crucial for making the framework accessible to a broader audience, ensuring it supports both in-depth analysis and clear, concise communication.

The abundance of information, while beneficial for users who require in-depth data analysis, may be excessive for those who need to only focus on the most relevant points. While it is recommended to make the description detailed since visually impaired individuals can process auditory information quickly, there are no specific guidelines on how long the description should be. This suggests a potential need for guidelines that help users adjust the level of detail to meet particular standards. In the absence of such guidelines, users are left to rely on their own judgment to determine which details are necessary and which can be omitted, potentially leading to inconsistent experiences.

5.2.2 Lack of Standardized Accessibility Guidelines

The effectiveness of the framework heavily depends on the specific requirements set by the accessibility department. While it is suggested that a description should range from two to eight sentences [47], no standardized guidelines are currently in place. Participants expressed that if the expectation is to provide only brief overviews, the tool might be perceived as overly complex. On the other hand, if more detailed descriptions are required, the tool could serve as a valuable resource, particularly for those who are new to the process. However, there is a concern that as users become more familiar with writing such descriptions, they may find the framework less useful over time. This raises the possibility that the framework's long-term value could diminish for experienced users unless it evolves to address their changing needs.

5.2.3 Time-Saving Efficiency

There are mixed opinions among participants when evaluating the framework's efficiency in saving time. While some view it as one of the primary strengths, others express concerns about its practicality in real-world scenarios. The concerns mainly stem from the design of the authoring interface and the volume of information that needs to be managed, which some participants feel could potentially complicate rather than streamline the description creation process.

The design of the Home page is generally well-received, particularly for its organized layout of three interactive views: the list of features, the description section, and the input visualization. However, the process of rephrasing and rearranging story slices, coupled with the need to switch between the main page and the Export page introduces inefficiencies. These steps could potentially disrupt the flow and diminish the overall user experience, making the interface less intuitive and more cumbersome to navigate.

Another concern is how to persuade users, especially journalists working in high-pressure, fast-paced newsrooms where speed is critical, to adopt the tool. The extensive information provided, although comprehensive, may be overwhelming for those unfamiliar with how to select specific features that should be included in the description. There is a fear that, instead of streamlining the process, the framework slows down users who are trying to figure out what to include, making it potentially more time-consuming than manually writing descriptions, even if those descriptions are less detailed or imperfect.

5.2.4 Data-Driven Narrative

The framework offers detailed insights into the underlying data but falls short in guiding users toward high-level communicative objectives [94], such as raising awareness about a particular issue or convincing and influencing the audience to adopt a specific viewpoint. In journalistic contexts, the primary goal is often to tell a story that resonates with a general audience. This requires summarizing story slices and telling the overall meaning and context of the data that capture the trends rather than focusing on specific details or granular data points. The framework's current design, which leans heavily on data detail, may not adequately support summarizing the story behind the visualization.

Another limitation is that visualizations serve both communicative and explorative purposes, as balancing these dual roles is crucial for effective data presentation [95]. However, as the framework relies on users to produce descriptions, the resulting narratives may end up being solely communicative as they inherently reflect their own perspectives. Determining whether a textual description can be explorative remains an open question. However, the explorative nature of visualization is closely tied to the amount of information it presents and Bertin's reading levels taxonomy offers a useful framework for evaluating information depth [96]. This taxonomy identifies three levels: 1) whole: the user can ask questions about all the data; 2) group: the user can ask questions about a subgroup of data; and 3) single: the user can only study single items. This taxonomy can also be applied to assess the level of information in a description. However, our framework only provides guidance on what information to include, without enforcing strict rules on the content of the description.

Additionally, the process of piecing together numerous detailed story slices to form a coherent description can be cumbersome. Users may find that the effort required to assemble multiple pieces of information outweighs the benefits, leading them to prefer writing their own descriptions from scratch. While large language models can be used to summarize narrative segments, the resulting summaries may obscure the meaning of individual sentences and make it difficult to connect them back to the original visualization through annotations. Some research prototypes have attempted to match captions with charts by highlighting visually prominent regions [97]. However, the accuracy of these methods cannot be guaranteed, and no standardized baseline for evaluating their effectiveness has been established.

Although the framework is effective in delivering detailed data insights, it needs enhancements to better support the creation of meaningful narratives. These enhancements should focus on improving its ability to summarize trends and present data insights in an accessible and engaging manner, ultimately bridging the gap between data exploration and communication.

5.3 Future Research

Feedback from the user study reveals that the descriptions generated by the framework are frequently regarded as overly scientific, making them less accessible to general users. To enhance usability and ensure broader appeal, the descriptions should be refined by simplifying the language, focusing on high-level insights rather than in-depth technical details. This could involve improving template wording [98] or integrating large language models with proper prompts [99] to produce more user-friendly content. Additionally, the framework should place greater emphasis on overall trends over individual data points using a taxonomy of visually prominent regions [100] and quantifying trend changes in greater detail, thereby providing a more nuanced interpretation of magnitude.

Regarding the list of features, although some participants found it overly complex, the inclusion of chart signatures and data sources in descriptions was recommended to improve transparency and credibility. To simplify the framework for general users, we propose improving it by automatically recommending top insights [101], with these checkboxes pre-selected by default. This approach would streamline the user experience by emphasizing the most relevant information, reducing the need for users to sift through a long list of features.

Expanding beyond the current scope, ongoing research into accessibility remains crucial, as preferences for description length and detail vary widely among users [102]. The absence of precise guidelines for accessibility introduces uncertainty in determining the optimal amount of information in textual descriptions. This ongoing challenge highlights the need to conduct further user studies with visually impaired individuals, as there is still a substantial gap in understanding the diverse motivations of people with various types of disabilities [103]. Gaining insights into their specific needs and preferences is essential for improving the accessibility of visualizations [104, 105]. Through feedback from visually impaired people, we can identify particular challenges they face when interpreting chart descriptions and explore ways to adapt the framework to better serve this community.

REFERENCES

[1] E. Segel and J. Heer, "Narrative visualization: Telling stories with data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1139–1148, 2010.

[2] A. Sharif *et al.*, "Understanding screen-reader users' experiences with online data visualizations," in *Proceedings of the 23rd International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS '21. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: https://doi.org/10.1145/3441852.3471202

[3] "Blindness and vision impairment," World Health Organization. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment

[4] J. Pal, U. Vallauri, and V. Tsaran, "Low-cost assistive technology in the developing world: a research agenda for information schools," in *Proceedings of the 2011 IConference*, ser. iConference '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 459–465. [Online]. Available: https://doi.org/10.1145/1940761.1940824

[5] "Html elements reference," Mozilla. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTML/Element

[6] "Screen reader user survey 10 results," WebAIM. [Online]. Available: https://webaim.org/projects/screenreadersurvey10/

[7] "H37: Using alt attributes on img elements," W3C. [Online]. Available: https://www.w3.org/TR/WCAG20-TECHS/H37.html

[8] "H45: Using longdesc," W3C. [Online]. Available: https://www.w3.org/TR/WCAG20-TECHS/H45.html

[9] V. L. Hanson and J. T. Richards, "Progress on website accessibility?" *ACM Trans. Web*, vol. 7, no. 1, mar 2013. [Online]. Available: https://doi.org/10.1145/2435215.2435217

[10] J. Snyder, "Audio description: The visual made verbal," *International Congress Series*, vol. 1282, pp. 935–939, 2005, vision 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0531513105009817

[11] S. Rai, J. Greening, and L. Petré, “A comparative study of audio description guidelines prevalent in different countries,” *Londra: Royal National Institute of Blind People*, 2010.

[12] I. Mazur and A. Chmiel, “Audio description training: A snapshot of the current practices,” *The Interpreter and Translator Trainer*, vol. 15, no. 1, pp. 51–65, 2021. [Online]. Available: https://doi.org/10.1080/1750399X.2021.1880263

[13] A. Matamala and P. Orero, “Designing a course on audio description and defining the main competences of the future professional,” *Linguistica Antverpiensia, New Series – Themes in Translation Studies*, vol. 6, Oct. 2021. [Online]. Available: https://lans-tts.uantwerpen.be/index.php/LANS-TTS/article/view/195

[14] K. Marriott *et al.*, “Inclusive data visualization for people with disabilities: a call to action,” *Interactions*, vol. 28, no. 3, p. 47–51, apr 2021. [Online]. Available: https://doi.org/10.1145/3457875

[15] B. Kasdorf, “Why accessibility is hard and how to make it easier: Lessons from publishers,” *Learned Publishing*, vol. 31, no. 1, pp. 11–18, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/leap.1146

[16] J. Boy *et al.*, “A principled way of assessing visualization literacy,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1963–1972, 2014.

[17] V. S. Morash *et al.*, “Guiding novice web workers in making image descriptions using templates,” *ACM Trans. Access. Comput.*, vol. 7, no. 4, nov 2015. [Online]. Available: https://doi.org/10.1145/2764916

[18] C. Chen *et al.*, “Figure captioning with relation maps for reasoning,” in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 1526–1534.

[19] J. Obeid and E. Hoque, “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*, B. Davis *et al.*, Eds. Dublin, Ireland: Association for Computational Linguistics, Dec. 2020, pp. 138–147. [Online]. Available: https://aclanthology.org/2020.inlg-1.20

[20] S. Kantharaj *et al.*, “Chart-to-text: A large-scale benchmark for chart summarization,” in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, S. Muresan, P. Nakov, and A. Villavicencio,

Eds. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 4005–4023. [Online]. Available: https://aclanthology.org/2022.acl-long.277

[21] Z.-Q. Cheng, Q. Dai, and A. G. Hauptmann, "Chartreader: A unified framework for chart derendering and comprehension without heuristic rules," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 22 202–22 213.

[22] "Accessibility principles," World Wide Web Consortium. [Online]. Available: https://www.w3.org/WAI/fundamentals/accessibility-principles/

[23] "Vega-lite – a grammar of interactive graphics," GitHub. [Online]. Available: https://vega.github.io/vega-lite/

[24] "Highcharts – interactive charting library for developers," Highcharts. [Online]. Available: https://www.highcharts.com/

[25] F. Elavsky, C. Bennett, and D. Moritz, "How accessible is my visualization? evaluating visualization accessibility with chartability," *Computer Graphics Forum*, vol. 41, no. 3, pp. 57–70, 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14522

[26] M. R. Morris *et al.*, "Rich representations of visual content for screen reader users," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, ser. CHI '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1–11. [Online]. Available: https://doi.org/10.1145/3173574.3173633

[27] J. R. Thompson *et al.*, "Chart reader: Accessible visualization experiences designed with screen reader users," ser. CHI '23. New York, NY, USA: Association for Computing Machinery, 2023. [Online]. Available: https://doi.org/10.1145/3544548.3581186

[28] T. Ferro and D. Pawluk, "Providing dynamic access to electronic tactile diagrams," in *Universal Access in Human–Computer Interaction. Designing Novel Interactions*, M. Antona and C. Stephanidis, Eds. Cham: Springer International Publishing, 2017, pp. 269–282.

[29] A. L. Corn and R. S. Wall, "Training and availability of braille transcribers in the united states," *Journal of Visual Impairment & Blindness*, vol. 96, no. 4, pp. 223–232, 2002. [Online]. Available: https://doi.org/10.1177/0145482X0209600404

[30] M. Mukhiddinov and S.-Y. Kim, "A systematic literature review on the automatic creation of tactile graphics for the blind and visually impaired," *Processes*, vol. 9, no. 10, 2021. [Online]. Available: https://www.mdpi.com/2227-9717/9/10/1726

[31] R. E. Ladner *et al.*, "Automating tactile graphics translation," in *Proceedings of the 7th International ACM SIGACCESS Conference on Computers and Accessibility*, ser. Assets '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 150–157. [Online]. Available: https://doi.org/10.1145/1090785.1090814

[32] C. Jayant *et al.*, "Automated tactile graphics translation: in the field," in *Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility*, ser. Assets '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 75–82. [Online]. Available: https://doi.org/10.1145/1296843.1296858

[33] C. Brown and A. Hurst, "Viztouch: automatically generated tactile visualizations of coordinate spaces," in *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*, ser. TEI '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 131–138. [Online]. Available: https://doi.org/10.1145/2148131.2148160

[34] R. Štampach and E. Mulíčková, "Automated generation of tactile maps," *Journal of Maps*, vol. 12, no. sup1, pp. 532–540, 2016. [Online]. Available: https://doi.org/10.1080/17445647.2016.1196622

[35] D. Guinness *et al.*, "Robographics: Dynamic tactile graphics powered by mobile robots," in *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 318–328. [Online]. Available: https://doi.org/10.1145/3308561.3353804

[36] B. O. D. P. Anke M. Brock, Philippe Truillet and C. Jouffrais, "Interactivity improves usability of geographic maps for visually impaired people," *Human–Computer Interaction*, vol. 30, no. 2, pp. 156–194, 2015. [Online]. Available: https://doi.org/10.1080/07370024.2014.924412

[37] J. A. Miele, S. Landau, and D. Gilden, "Talking tmap: Automated generation of audio-tactile maps using smith-kettlewell's tmap software," *British Journal of Visual Impairment*, vol. 24, no. 2, pp. 93–100, 2006. [Online]. Available: https://doi.org/10.1177/0264619606064436

[38] C. M. Baker *et al.*, "Tactile graphics with a voice: using qr codes to access text in tactile graphics," in *Proceedings of the 16th International ACM SIGACCESS Conference on Computers & Accessibility*, ser. ASSETS '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 75–82. [Online]. Available: https://doi.org/10.1145/2661334.2661366

[39] G. Fusco and V. S. Morash, "The tactile graphics helper: Providing audio clarification for tactile graphics using machine vision," ser. ASSETS '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 97–106. [Online]. Available: https://doi.org/10.1145/2700648.2809868

[40] W. Yu, R. Ramloll, and S. A. Brewster, "Haptic graphs for blind computer users," in *Proceedings of the First International Workshop on Haptic Human-Computer Interaction.* Berlin, Heidelberg: Springer-Verlag, 2000, p. 41–51.

[41] R. G. G. Matt Rice, R. Daniel Jacobson and D. Jones, "Design considerations for haptic and auditory map interfaces," *Cartography and Geographic Information Science*, vol. 32, no. 4, pp. 381–391, 2005. [Online]. Available: https://doi.org/10.1559/152304005775194656

[42] L. Zeng and G. Weber, "Audio-haptic browser for a geographical information system," in *Computers Helping People with Special Needs*, K. Miesenberger *et al.*, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 466–473.

[43] H. Takagi *et al.*, "Analysis of navigability of web applications for improving blind usability," vol. 14, no. 3, p. 13–es, sep 2007. [Online]. Available: https://doi.org/10.1145/1279700.1279703

[44] V. Potluri *et al.*, "Examining visual semantic understanding in blind and low-vision technology users," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, ser. CHI '21. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: https://doi.org/10.1145/3411764.3445040

[45] K. Williams *et al.*, "Find and seek: Assessing the impact of table navigation on information look-up with a screen reader," vol. 12, no. 3, aug 2019. [Online]. Available: https://doi.org/10.1145/3342282

[46] J. Zong *et al.*, "Rich screen reader experiences for accessible data visualization," *Computer Graphics Forum*, vol. 41, no. 3, pp. 15–27, 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14519

[47] C. Jung *et al.*, “Communicating visualizations without visuals: Investigation of visualization alternative text for people with visual impairments,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 1, pp. 1095–1105, 2022.

[48] “Charts - google for developers,” Google. [Online]. Available: https://developers.google.com/chart

[49] S. E. Kahou *et al.*, “Figureqa: An annotated figure dataset for visual reasoning,” 2018. [Online]. Available: https://arxiv.org/abs/1710.07300

[50] K. Kafle *et al.*, “Dvqa: Understanding data visualizations via question answering,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5648–5656.

[51] ——, “Answering questions about data visualizations using efficient bimodal fusion,” in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 1487–1496.

[52] A. Strantz, “Beyond “alt-text”: Creating accessible data visualizations with code,” in *Proceedings of the 39th ACM International Conference on Design of Communication*, ser. SIGDOC ’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 331–337. [Online]. Available: https://doi.org/10.1145/3472714.3473661

[53] D. H. Kim, E. Hoque, and M. Agrawala, “Answering questions about charts and generating visual explanations,” in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, ser. CHI ’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1–13. [Online]. Available: https://doi.org/10.1145/3313831.3376467

[54] A. Sharif *et al.*, “Voxlens: Making online data visualizations accessible with an interactive javascript plug-in,” in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, ser. CHI ’22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3491102.3517431

[55] “Sonification report,” ICAD. [Online]. Available: https://www.icad.org/websiteV2.0/References/nsf.html

[56] L. Rubinstein and E. M. Gruenberg, “Intramodal and crossmodal sensory transfer of visual and auditory temporal patterns,” *Perception & Psychophysics*, vol. 9, no. 5, pp. 385–390, 1971.

[57] S. E. Guttman, L. A. Gilroy, and R. Blake, “Hearing what the eyes see: Auditory encoding of visual temporal sequences,” *Psychological Science*, vol. 16, no. 3, pp. 228–235, 2005. [Online]. Available: http://www.jstor.org/stable/40064206

[58] K. Enge *et al.*, “Open your ears and take a look: A state-of-the-art report on the integration of sonification and visualization,” *Computer Graphics Forum*, vol. 43, no. 3, p. e15114, 2024. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.15114

[59] “Audio graphs,” Apple. [Online]. Available: https://developer.apple.com/documentation/accessibility/audio-graphs

[60] S. Cantrell, B. Walker, and Moseng, “Highcharts sonification studio: An online, open-source, extensible, and accessible data sonification tool,” 06 2021, pp. 210–216.

[61] J. H. Howard and J. A. Ballas, “Acquisition of acoustic pattern categories by exemplar observation,” *Organizational Behavior and Human Performance*, vol. 30, no. 2, pp. 157–173, 1982. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0030507382902161

[62] A. Lundgard and A. Satyanarayan, “Accessible visualization via natural language descriptions: A four-level model of semantic content,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 1, pp. 1073–1083, 2022.

[63] A. Mahinpei, Z. Kostic, and C. Tanner, “Linecap: Line charts for data visualization captioning models,” in *2022 IEEE Visualization and Visual Analytics (VIS)*, 2022, pp. 35–39.

[64] D. Guinness, E. Cutrell, and M. R. Morris, “Caption crawler: Enabling reusable alternative text descriptions using reverse image search,” in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, ser. CHI ’18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1–11. [Online]. Available: https://doi.org/10.1145/3173574.3174092

[65] C. Gleason *et al.*, “Twitter a11y: A browser extension to make twitter images accessible,” in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, ser. CHI ’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1–12. [Online]. Available: https://doi.org/10.1145/3313831.3376728

[66] M. R. Zhang, M. Zhong, and J. O. Wobbrock, “Ga11y: An automated gif annotation system for visually impaired users,” in *Proceedings of the 2022*

CHI Conference on Human Factors in Computing Systems, ser. CHI ’22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3491102.3502092

[67] J. Luo *et al.*, “Chartocr: Data extraction from charts images via a deep hybrid framework,” in *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2021, pp. 1916–1924.

[68] L. Gong, J. Crego, and J. Senellart, “Enhanced transformer model for data-to-text generation,” in *Proceedings of the 3rd Workshop on Neural Generation and Translation*, A. Birch *et al.*, Eds. Hong Kong: Association for Computational Linguistics, Nov. 2019, pp. 148–156. [Online]. Available: https://aclanthology.org/D19-5615

[69] A. Spreafico and G. Carenini, “Neural data-driven captioning of time-series line charts,” in *Proceedings of the 2020 International Conference on Advanced Visual Interfaces*, ser. AVI ’20. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3399715.3399829

[70] K. Papineni *et al.*, “Bleu: a method for automatic evaluation of machine translation,” ser. ACL ’02. USA: Association for Computational Linguistics, 2002, p. 311–318. [Online]. Available: https://doi.org/10.3115/1073083.1073135

[71] C.-Y. Lin, “ROUGE: A package for automatic evaluation of summaries,” in *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 74–81. [Online]. Available: https://aclanthology.org/W04-1013

[72] Z. Cui *et al.*, “Datasite: Proactive visual data exploration with computation of insight-based recommendations,” *Information Visualization*, vol. 18, no. 2, pp. 251–267, 2019. [Online]. Available: https://doi.org/10.1177/1473871618806555

[73] Y. Wang *et al.*, “Datashot: Automatic generation of fact sheets from tabular data,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 895–905, 2020.

[74] C. Liu *et al.*, “Autocaption: An approach to generate natural language description from visualization automatically,” in *2020 IEEE Pacific Visualization Symposium (PacificVis)*, 2020, pp. 191–195.

[75] I. Sahin, “Detailed review of rogers’ diffusion of innovations theory and educational technology-related studies based on rogers’ theory.” *Turkish Online*

Journal of Educational Technology, vol. 5, pp. 14–23, 2006. [Online]. Available: https://api.semanticscholar.org/CorpusID:2289546

[76] E. Salisbury, E. Kamar, and M. Morris, "Toward scalable social alt text: Conversational crowdsourcing as a tool for refining vision-to-language technology for the blind," *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, vol. 5, no. 1, pp. 147–156, Sep. 2017. [Online]. Available: https://ojs.aaai.org/index.php/HCOMP/article/view/13301

[77] A. Dasgupta *et al.*, "Familiarity vs trust: A comparative study of domain scientists' trust in visual analytics and conventional analysis methods," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 271–280, 2017.

[78] K. Mack *et al.*, "Designing tools for high-quality alt text authoring," in *Proceedings of the 23rd International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS '21. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: https://doi.org/10.1145/3441852.3471207

[79] R. Amar, J. Eagan, and J. Stasko, "Low-level components of analytic activity in information visualization," in *Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization*, ser. INFOVIS '05. USA: IEEE Computer Society, 2005, p. 15. [Online]. Available: https://doi.org/10.1109/INFOVIS.2005.24

[80] B. Saket, A. Endert, and Demiralp, "Task-based effectiveness of basic visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 7, pp. 2505–2512, 2019.

[81] G. J. Quadri and P. Rosen, "A survey of perception-based visualization studies by task," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 12, pp. 5026–5048, 2022.

[82] C. Xiong *et al.*, "Visual arrangements of bar charts influence comparisons in viewer takeaways," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 1, pp. 955–965, 2022.

[83] S. Latif, D. Liu, and F. Beck, "Exploring interactive linking between text and visualization," in *Proceedings of the Eurographics/IEEE VGTC Conference on Visualization: Short Papers*, ser. EuroVis '18. Goslar, DEU: Eurographics Association, 2018, p. 91–94.

[84] A. Srinivasan *et al.*, "Augmenting visualizations with interactive data facts to facilitate interpretation and communication," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 672–681, 2019.

[85] "React," Meta Open Source. [Online]. Available: https://react.dev/

[86] "Flask," Flask. [Online]. Available: https://flask.palletsprojects.com/en/3.0.x/

[87] "Visualization types," Datawrapper. [Online]. Available: https://developer.datawrapper.de/docs/chart-types

[88] M. R. Luo, *CIELAB*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 1–7. [Online]. Available: https://doi.org/10.1007/978-3-642-27851-8_11-1

[89] K. Taunk *et al.*, "A brief review of nearest neighbor algorithm for learning and classification," in *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, 2019, pp. 1255–1260.

[90] H. Vinutha, B. Poornima, and B. Sagar, "Detection of outliers using interquartile range technique from intrusion dataset," in *Proceedings of the International Conference on Frontiers of Intelligent Computing*. Springer, 2018, pp. 511–518.

[91] J. Nielsen, "Estimating the number of subjects needed for a thinking aloud test," *Int. J. Hum.-Comput. Stud.*, vol. 41, no. 3, p. 385–397, sep 1994. [Online]. Available: https://doi.org/10.1006/ijhc.1994.1065

[92] J. Nielsen and R. Molich, "Heuristic evaluation of user interfaces," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '90. New York, NY, USA: Association for Computing Machinery, 1990, p. 249–256. [Online]. Available: https://doi.org/10.1145/97243.97281

[93] C. Stokes *et al.*, "Striking a balance: Reader takeaways and preferences when integrating text and charts," *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 1, pp. 1233–1243, 2023.

[94] E. Adar and E. Lee, "Communicative visualizations as a learning problem," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 946–956, 2021.

[95] J. Hullman and N. Diakopoulos, "Visualization rhetoric: Framing effects in narrative visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2231–2240, 2011.

[96] J. Bertin, *Semiology of graphics.* University of Wisconsin Press, 1983.

[97] D. H. Kim *et al.*, "Emphasischecker: A tool for guiding chart and caption emphasis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 30, no. 1, pp. 120–130, 2024.

[98] C. L. Teles De Oliveira *et al.*, "Proposal and evaluation of textual description templates for bar charts vocalization," in *2019 23rd International Conference Information Visualisation (IV)*, 2019, pp. 163–169.

[99] D. Bromley and V. Setlur, "What is the difference between a mountain and a molehill? quantifying semantic labeling of visual features in line charts," in *2023 IEEE Visualization and Visual Analytics (VIS)*, 2023, pp. 161–165.

[100] D. H. Kim, V. Setlur, and M. Agrawala, "Towards understanding how readers integrate charts and captions: A case study with line charts," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, ser. CHI '21. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: https://doi.org/10.1145/3411764.3445443

[101] B. Tang *et al.*, "Extracting top-k insights from multi-dimensional data," in *Proceedings of the 2017 ACM International Conference on Management of Data*, ser. SIGMOD '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1509–1524. [Online]. Available: https://doi.org/10.1145/3035918.3035922

[102] A. Stangl, M. R. Morris, and D. Gurari, ""person, shoes, tree. is the person naked?" what people with vision impairments want in image descriptions," ser. CHI '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1–13. [Online]. Available: https://doi.org/10.1145/3313831.3376404

[103] N. W. Kim *et al.*, "Accessible visualization: Design space, opportunities, and challenges," *Computer Graphics Forum*, vol. 40, no. 3, pp. 173–188, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14298

[104] A. Lundgard, C. Lee, and A. Satyanarayan, "Sociotechnical considerations for accessible visualization design," in *2019 IEEE Visualization Conference (VIS)*, 2019, pp. 16–20.

[105] K. Angerbauer and M. Sedlmair, "Toward inclusion and accessibility in visualization research: Speculations on challenges, solution strategies, and calls for action (position paper)," in *2022 IEEE Evaluation and Beyond - Methodological Approaches for Visualization (BELIV)*, 2022, pp. 20–27.

APPENDIX A   ANNOTATION TECHNIQUES

Table A.1 Feature-specific annotations based on chart type

	d3-bars	d3-bars-split	d3-bars-stacked	d3-bars-grouped	column-chart	grouped-column-chart	stacked-column-chart	d3-area	d3-lines	d3-pies
type	Change in background color									
title, subtitle, footnote	Highlighting text									
axes	Highlighting the x or y axis									N/A
color	Textured pattern on referenced bars, columns, or areas								Highlighting lines	Textured pattern
extrema	Arrows with labels	Textured pattern on referenced bars	Arrows with labels (stacked values), textured pattern (within or across categories)	Vertical lines with labels (within categories), textured pattern (across categories)	Arrows with labels	Horizontal lines with labels (within categories), textured pattern (across categories)	Arrows with labels (stacked values), texture pattern (within or across categories)	Arrows with labels (stacked values and within categories)	Arrows with labels (within categories)	Textured pattern on referenced slices
mean and median	Vertical line with label	Vertical lines with labels (within categories)	Vertical line with label (stacked values)	Vertical lines with labels (within categories), shorter lines (across categories)	Horizontal line with label	Horizontal lines with labels (within categories), shorter lines (across categories)	Horizontal line with label (stacked values)	Horizontal lines with labels (stacked values and within categories)	Horizontal lines with labels (within categories)	N/A
outliers	Arrows with labels	Textured pattern on referenced bars	Arrows with labels (stacked values), textured pattern (within or across categories)	Arrows with labels (within categories)	Arrows with labels	Arrows with labels (within categories)	Arrows with labels (stacked values), textured pattern (within or across categories)	Arrows with labels (stacked values and within categories)	Arrows with labels (within categories)	Textured pattern on referenced slices
correlation (category-wise)	None	N/A	Textured pattern (stacked values)	Textured pattern	None	Textured pattern	Textured pattern (stacked values)	Textured pattern (stacked values and within categories)	Highlighting referenced lines	N/A
trend (category-wise)	None	N/A	Textured pattern (stacked values)	Textured pattern	Highlighting areas of increase and decrease	Textured pattern (whole trends) or highlighting areas of increase and decrease	Textured pattern (whole trends) or highlighting areas of increase and decrease	Textured pattern (whole trends) or highlighting areas of increase and decrease	Highlighting referenced lines (whole trends) or the areas of increase and decrease	N/A

APPENDIX B INTERVIEW QUESTIONS

Optional

1. Have you tried to import any chart using the token? How many charts did you import?

Usefulness

1. In your view, how well does the tool help users to recognize the important insights in a chart?
2. Can you share your thoughts on how the tool's design aids in communicating insights? Did the annotations on the chart help you to interpret the findings in the chart?
3. Do you think the tool helps you write a description that contains all the necessary parts? Are there any features you believe could be improved to enhance this?
4. Is there anything that particularly stood out to you, either positively or negatively? Do you think the features/views/functions below are efficient?
    - selection of a chart
    - list of important elements
    - respective descriptions for selected elements
    - descriptions that can be edited and reordered
    - visual cues on the input chart
5. If you have to write a description, do you think the tool can save you time?
6. Do you think the tool improves efficiency in writing descriptions for charts?

Usability

1. Do you think the tool is easy to use? Do you think the tool is efficient?
2. Is the tool easy to learn? Are there any parts that are hard to understand?
3. Did you encounter any difficulties while using the tool? Are there any aspects of the tool that can be optimized to make it easier for you to use?

4. Based on your experience, do you feel that the tool can be used by a more general audience, in addition to chart creators, including those without advanced expertise?

Follow-up Questions

1. Is there anything you would like to say about this project from a journalism standpoint?
2. Are there any features or functionalities you wish the tool had but currently lacks?

APPENDIX C ONLINE SURVEY

Usability

1. I can easily understand how to use the tool
2. I can learn how to use the tool quickly
3. Selecting or importing a chart is straightforward
4. I can easily understand the purpose of the checklist on the left
5. The interactions between the checklist and the descriptions in the middle are easy to understand
6. The annotations on the chart can effectively capture my attention and enhance my information processing
7. When the input is supported, the tool works as expected
8. The user interface is easy to understand
9. Interactions with the tool are clear and comprehensible
10. The tool's usage is straightforward and manageable
11. Overall, the tool is user-friendly

Efficacy

1. Using the tool is more efficient compared to reading a list of accessibility guidelines for describing charts
2. The tool makes the task of writing descriptions easier
3. The tool can help me avoid omitting details when writing descriptions
4. The tool can help me avoid making mistakes when writing about specific values
5. If I have to describe charts I created, the tool can save me time
6. If I have to describe charts I created, the tool can help me improve my job performance

Utility

1. The tool can be integrated into my existing workflow without changing too much my habit
2. The checklist on the left helps me to know all the components of an input chart
3. Generated descriptions help me to recognize all essential insights in the input chart
4. The tool offers helpful guidance in writing descriptions
5. The interactions on the generated descriptions (edit and reorder) help in refining the description
6. Essential features or functionalities for writing descriptions are supported by the tool
7. The completeness of my descriptions is improved by using the tool
8. I find the tool valuable and useful
9. I am satisfied with the tool
10. I would recommend the tool to other professionals

Demographic Survey

1. What is your gender? [Male, Female, Other]
2. What is your age?
3. How many years of experience in journalism do you have?
4. What is your expertise level with Datawrapper? [Novice, Advanced Beginner, Competent, Proficient, Expert]
5. On average, how many charts do you create with Datawrapper in one month?
6. Have you ever described the visualization that you created?