Eurographics Conference on Visualization (EuroVis) 2018
J. Heer, H. Leitte, and T. Ropinski
(Guest Editors)

Volume 37 (2018), Number 3

ChangeCatcher: Increasing Inter-author Awareness for Visualization Development

M. Loorak[1], M. Tory[2] and S. Carpendale[3]

[1]Autodesk Research, Canada [2]Tableau Research, USA [3]University of Calgary, Canada

Abstract

We introduce an approach for explicitly revealing changes between versions of a visualization workbook to support version comparison tasks. Visualization authors may need to understand version changes for a variety of reasons, analogous to document editing. An author who has been away for a while may need to catch up on the changes made by their co-author, or a person responsible for formatting compliance may need to check formatting changes that occurred since the last time they reviewed the work. We introduce ChangeCatcher, a prototype tool to help people find and understand changes in a visualization workbook, specifically, a Tableau workbook. Our design is based on interviews we conducted with experts to investigate user needs and practices around version comparison. ChangeCatcher provides an overview of changes across six categories, and employs a multi-level details-on-demand approach to progressively reveal details. Our qualitative study showed that ChangeCatcher's methods for explicitly revealing and categorizing version changes were helpful in version comparison tasks.

Categories and Subject Descriptors (according to ACM CCS): H.5.2 [Information Interfaces and Presentation]: User Interfaces—Screen design

1. Introduction

Understanding and communicating changes is a known problem in asynchronous collaboration across many domains including text editing [HM76, NCK*92], sense making [ZGI*18], programming [Tic85, LM12], and tracking patient status [AKP14]. This problem also occurs in asynchronous visualization authoring, where authors may take turns editing, build on top of one another's work, or review visualizations created by a colleague. In these situations, analysts are faced with the initial task of identifying what a colleague has changed in the visualizations. This becomes more challenging in Business Intelligence (BI) tools such as Tableau [TAB], Qlik [QLI], and Power BI [MIC], in which multiple visualizations are grouped in a file that we will call a visualization *workbook*. We explore the challenge of identifying and communicating visualization changes through the design, development, and evaluation of ChangeCatcher, a tool that reveals visualization version differences for asynchronous collaboration scenarios.

Our work is motivated by eight exploratory interviews with professional analysts around their collaboration practices. Participants reported using two methods to manage version changes in their collaborative work, both of which are cumbersome: manual documentation and side-by-side comparison. For example, in a collaborative data story project, a visualization designer reported posting each new version (in an online collaboration tool) for review and feedback by other designers and a domain expert. These posts typically included a lengthy, detailed explanation of all the changes made since the last version. In addition to relying on written documentation of the changes, interviewees identified changes between versions of a visualization workbook via side-by-side manual inspection. While this current practice does work to some degree, it is cumbersome and error prone. Reviewers are likely to notice changes that dramatically influence the look of a visualization, but miss non-obvious ones that may be equally (or more) important, such as adjustments to a calculation formula or an aggregation function.

To help with such review tasks, we designed ChangeCatcher, a prototype that reveals the changes between versions of a visualization workbook. Change tracking tools for textual data cannot be applied to visualizations because of their non-linear and visual structure. Moreover, design requirements for visualization-specific change tracking are not well-explored. From our interviews with experts, we derived an understanding of the challenges analysts face when trying to track or discover changes in a visualization workbook. These challenges fall into two main categories: 1) discovering the changes between versions, and 2) reasoning about the changes. In this research, we focus on addressing the first category as it is the essential first step in any change review scenario.

ChangeCatcher uses Tableau workbooks as an exemplar, using detail-on-demand to reveal the changes. It aims to reduce the burden of finding differences, make change review more efficient, and reduce the likelihood of missing non-obvious changes. ChangeCatcher is primarily designed to support asynchronous co-authoring scenarios involving *handoff*, or the transfer of responsibility for a task from one author to another. However, several of our participants also envisioned using ChangeCatcher in single-user authoring scenarios. While our prototype is designed around Tableau workbooks, the principles are not limited to Tableau; the design ideas could readily be adapted to other tabular data visualization tools.

In our experiential study, 11 Tableau specialists used ChangeCatcher to perform a series of change discovery tasks. Results suggested that explicitly revealing visualization changes can make it easier to find changes, generate confidence that changes have been identified, and assist people in assessing work progress. Designing ChangeCatcher led us to re-examine the broader challenge of visualization versioning, which is quite different from version management in other domains such as document editing and software development. Observing and interviewing people about current version comparison practices deepened our understanding of the challenges in this space and revealed new use case scenarios.

In addition to the tool itself, we 1) contribute scenarios and design requirements for version comparison in visualization, including important functionality and categories of change types, 2) demonstrate the value of explicitly revealing visualization changes, and 3) establish that understanding visualization changes is an important task in asynchronous collaboration, unique from version comparison in other domains. Our results provide insight into the design of future change review tools for visualization workbooks.

2. Related Work

Since we could not find previous work directly about versioning and handoff for visualization, we review two closely related areas: work on supporting handoff tasks and work on comparative visualization.

2.1. Handoff in asynchronous collaboration

Handoff in asynchronous collaboration has been studied extensively in other fields, notably nursing and air traffic control, where shift changes result in fully transferring the workload between professionals [Sha08]. A systematic review of handoff studies in healthcare [AKP14] identified the quality of handoff as a critical factor in patient care, and noted efforts to improve handoff through standardized information transfer and communication practices. It is well known that poor quality handoffs can introduce errors and inefficiencies [Aus05], and that relying on human communication alone (discussions and documents) is often insufficient [AKP14, FSJC13].

While shift changes may be the most extreme case of handoff in collaborative work, handoffs also occur in a *referral* context. In a referral handoff, responsibility and task material are shared but the environment may differ [Sha08]. Typically referral handoffs involve different perspectives or skills [Sha08], like in our usage scenarios, where each collaborator brings design ideas or an expert reviews the design work of a colleague.

Handoff has been much less studied in visual analytics, but is a recognized problem. A review by Xu et al. [XAJK*15] called on the visual analytics community to study handoff and to learn best practices and information needs from other domains. Numerous authors have suggested that history and provenance tools could be useful for collaborative communication including handoff [HVHC*08, RESC16, XAJK*15]. However, most provenance tools (e.g., [HMSA08]) capture and replay the history of edits step-by-step, rather than supporting review of all changes from one version to the next. Such step-by-step views are at too detailed a level to be effective for handoff tasks, where the outcome is more important than the individual steps [LWPL11, PM09, ST15].

Asynchronous collaboration tools may also capture and share *externalizations* (recorded findings, hypotheses, etc.) that can provide awareness of a collaborator's work (e.g., [WHHA11, SvW08, PM09]). While useful, externalizations are insufficient for version sharing as they rely on time consuming and incomplete manual documentation. Studies on collaborative sensemaking found that efficient, low cost communication mechanisms are critical [Sha08]. Thus, several researchers have developed tools specific to handoff in sensemaking [PM09, ST15, Sha08], revealing questions asked, avenues of investigation, and findings. Our work focuses instead on collaborative *authoring*. Visualization authoring may involve sensemaking in its initial phases, but the overall focus is creation of a visualization to be used by others. Handoff may be for the collaborator to pick up the work (as in prior work on sensemaking) or for review and feedback. In this way, collaborative visualization authoring is similar to document authoring — what you typically do first is find out what changed since you last saw the document.

2.2. Visualizing differences

Tools for version comparison have a long history in Software Engineering [ESS92, EGK*01, LB85, VTvW05]. For instance, the classic *diff* algorithm [HM76] identifies line-by-line version differences in text documents. Software Engineering has also explored visual comparison of diagrams (e.g., [WDC03, MGH05, XS05]) that typically reveal added/deleted/changed items through color coding. These ideas were applied to visualization in VisTrails [FSC*06], where changes in a data flow program were shown via color. Visual comparison approaches also exist for slide presentations [PA06], storylines [TLZD16], and interface designs [HFR*10]. Experimental evidence shows that difference visualizations can reduce effort in some handoff tasks (e.g. [HFR*10]). However, this prior work has not considered visual diff tools for comparing visualization states.

More generally, there are numerous visualization approaches for comparison tasks. Gleicher et al. [GAW*11] identified three fundamental techniques. *Juxtaposition* designs present each object separately, either side-by-side (i.e. small multiples [Tuf91]) or temporally. *Superposition* designs overlay objects on top of one another. *Explicit representation* designs directly encode connections (e.g. directly visualizing the difference). Our own design is a hybrid of juxtaposition (of the workbook versions themselves) and explicit difference visualization (shown as an abstract representation). Gleicher et al. [GAW*11] report the advantage of this combination: explicit encodings help with understanding the connections between juxtaposed views, and the juxtaposed views help with understanding the differences in context.

3. Requirements Gathering

To understand current practices and challenges around collaborative visualization authoring, we conducted semi-structured interviews with eight professional analysts from outside our organizations. We focused on a past collaborative authoring project and asked about (1) roles of people involved, (2) tools used for analysis, visualization, and collaboration, and (3) collaboration challenges. Two groups also shared portions of their written communication channel (Slack in both cases). An open-coding analysis of the interview transcripts identified themes including: coordination and awareness strategies, collaboration platforms, concurrent editing, and separation of visualization and communication channels. A detailed discussion of all these themes is outside the scope of this publication; here we focus on findings related to change tracking.

Six out of eight participants mentioned change tracking; lack of change tracking or source control was often considered painful. Several participants reported regularly sharing a text description of their changes. Manually tracking changes was cumbersome, incomplete, and frustrating. [A] said, *"...it created more work. So when you have to write it all out it takes more time. And [. . .] the way I describe and the way you read it, we might just kind of miss each other."* [L] noted related problems, *"You've got a developer and you've got a test engineer. And the test engineer's job solely is to validate workbooks and make small changes. [. . .] where it becomes complicated is that the person that they handed off to may not understand what all of the differences are between the versions."* [J] described change management as a daily pain point, *"A dozen iterations or more, and then we'll finally land on something that ends up as the final project [. . .] I'm copying his sheet, making a modification to it, sending it back [. . .] I'm just telling you all the problems, but this is a constant, constant like daily pain point for me."*

Possible uses of change tracking included: co-authoring, collaborative review (feedback on another person's visualization), visualization testing/validation by someone other than the developer, historical review for compliance, and retrospective analysis of a visualization session for learning. Greater awareness, or earlier awareness, of changes made by others could also improve coordination and efficiency. [L] stated, *"we could have all been in tighter lockstep with each other [. . . If colleagues' work had been available earlier], we may have seen like some of the classification scheme that [they] came up with, and as a result of that shortened the time between making that change on our own."*

These interviews revealed the challenge of identifying what has changed in a visualization and motivated our work on ChangeCatcher. As far as we know, there is no coherent tool that fills this need by revealing version changes between visualizations.

4. Scenarios and Design Requirements

Here we articulate target use case scenarios and design requirements that drove the design of ChangeCatcher. We aimed to support the following asynchronous collaborative editing scenarios:

Co-authoring: Larry and Sarah are working together on a visualization dashboard that they will publish. After building a couple of draft dashboards, Larry gives the workbook to Sarah for her to edit and critique. Sarah makes some changes to improve the design and then adds more visualizations. Then, she sends it back to Larry. To effectively pick up on the work, Larry first needs understand what Sarah has changed.

Review: Jessica is responsible for reviewing dashboards her department develops to ensure conformity to formatting and style conventions. Bruce updated a published dashboard and Jessica checks the changes. Jessica wants to review updates only; which visualizations have changed and which changes are formatting?

Table 1: *Visualization Change Types.*

Change Type	Description
Encoding	Change to the mapping between data attributes and visual variables. In Tableau, these corresponded to changes in the shelf content.
Filtering	Changes to the included data records by restricting values on one or more dimensions.
Analytics	Changes to items that augment the chart (e.g., annotations, reference lines). In Tableau, these corresponded to items on the Analytics pane.
Arrange	Changing the ordering of items in the view: sorting, manual ordering, adding or deleting sheets from a dashboard, or changing positions of sheets in a dashboard.
Data	Changes involving operations on the dataset: editing calculated fields, the data hierarchy, or join operations.
Formatting	Superficial changes to the visual design, including titles, fonts, color choices, axis ticks/labels, and swapping the horizontal/vertical axes.

For co-authoring scenarios, we focus on supporting co-authors to identify the changes made by their collaborators, rather than to take action (*e. g.,* accept/reject) on the changes. While co-authoring scenarios involve direct editing by both people, review scenarios may involve a variety of roles. Besides formatting (as above), reviewers might provide feedback on content, visual design, encoding, or interaction. Reviewers might have greater expertise in visualization design than the primary author (design review by an expert), similar expertise (peer review), or less expertise (*e. g.,* review by a domain expert or end user). Our intent with ChangeCatcher was to provide a visualization of workbook changes to support task handoff in these collaborative editing scenarios. While ChangeCatcher might also be suitable for some single-user scenarios, this was not our focus.

Based on the target scenarios, interview findings, and an iterative ideation exercise, we articulated these design requirements:

DR1 **Reveal changes by sheet**: Users may be interested in changes only to a particular visualization/sheet. The system should support a high-level review of changes on a per-sheet basis so that irrelevant sheets can be ignored.

DR2 **Reveal changes by type**: Users may be interested only in particular type(s) of changes. For example, "*depending on my workbook, I would be concerned about different things. If I thought the data in my workbook was really good, I might be concerned when someone changed the data but if I had a workbook [without] good formatting, then I want to know about formatting changes*" [P02]. Similarly, a user might wish to confirm that a filter change was applied to all sheets, ignoring other changes. Therefore, the system should reveal changes on a per-type basis. We categorized changes into 6 types (Table 1). We developed the types iteratively, inspired by Brehmer and Munzner's task classification [BM13] and Heer et al.'s analysis of Tableau actions [HMSA08].

DR3 **Enable details-on-demand**: Since changes differ in importance depending on the user's goal, the system should provide an initial high level overview of all the version changes, and then allow drill-down into progressively more detail about each change.

DR4 **Seamless integration with workbook layout**: As visualization authors become intimately familiar with the design and layout of their workbooks, we wanted to avoid changing the workbook layout or using a completely new abstract design.

Our iterative design approach involved paper prototyping of many alternatives and gaining feedback from design experts to compare possibilities. For instance, one design approach we considered was visually overlaying the changes on top of the Tableau workbooks. This design has the benefits of being familiar and showing the changes within close proximity to their context. However, it can add clutter to the workbooks, it only shows changed elements in the current view, and it does not provide an overview of changes. Thus, the user still needs to go through the workbook sheet-by-sheet to find all the changes. Side-by-side comparison of views similarly requires going through the workbook sheet-by-sheet. We also considered several abstract views that were completely separated from the workbook, but determined that these might be difficult to relate to the workbook content. Our iterative approach led us to ChangeCatcher's hybrid design (Figure 1).

5. ChangeCatcher System Design

ChangeCatcher was designed to reveal differences between consecutive versions of a Tableau workbook. The prototype is implemented using D3 [BOH11]. Unlike visualization histories, we designed our system to reveal only differences between versions, not intermediate actions in the editing process. ChangeCatcher consists of a) *Overview Bars*, which reveal a high level overview of changes between versions, b) *Visualization Panel*, showing the contents of a selected sheet, and c) *Detail View*, an on-demand view that visually represents change details.

5.1. Overview Bars

Each overview bar represents a high-level synopsis of changes between two consecutive versions (see Figure 1). The bars are split into stacked segments, each representing a sheet with a change: blue represents a sheet/dashboard added in the newer version, orange represents deletion, and grey represents any other change. For consistency, we keep this color scheme throughout the rest of our designed visual elements. We chose stacked bar graphs for these high-level changes because they are familiar, can show different categories of changes plus their additive values, and use position as the most powerful visual cue [Mun14]. We included not only the most recent changes to the workbook but also the history of changes to earlier versions. For example, Figure 1 shows that between V3 and V4, two sheets were added, one was deleted, and four sheets changed. There were no changes between V2 and V3. Overview bars show the evolution of the whole workbook or individual sheets, *"I am interested to be able to compare the workbooks historically. [...] to see how the workbook changed over time"* [P05].

Hovering over a colored bar reveals details in a tooltip. The tooltip for a grey bar shows the number of sheets/dashboards that have changed (Figure 2), plus a list of the change types (see Table 1). This allows the analyst to understand the types of changes before deciding to dig deeper (**DR3**). For instance, if the analyst is only interested in the *Data* changes between V1 and V2, then reading the tooltip allows her to know whether any such changes exist. To examine the changes between two versions in detail, the user can press the expansion button (⛶) at the end of the bar. Pressing this button again will hide the detail view.

5.2. Detail View

Unlike textual data that are linear, visualization workbooks have a non-linear structure. It is essential to consider this difference when designing a system to reveal visualization changes. Overlaying changes on each sheet would cause occlusion and make it difficult to see an overview of changes by category (Table 1) (**DR3**).

Thus, to fulfill **DR3** and **DR4**, we augment the workbook with additional visual elements. This view consists of two ribbons of tabs: the lower one represents sheets/dashboards in the previous version and the upper ones are those in the newer version. For example, in Figure 1, the lower ribbon of tabs belongs to V4 and the upper one belongs to V5. In accordance with the color scheme in §5.1, blue tabs represent added sheets in V5, while orange ones have been deleted from V4. We draw lines between the tabs in the newer and older versions to explicitly visualize correspondence. To identify renamed tabs, we strikethrough the old name of the tab, a common way to indicate removed text (Figure 1).

As in Tableau workbooks, the tabs in the new version (V5) may go off screen if the number is large. Off-screen tabs are accessible in ChangeCatcher by dragging left or right. However, we included all the tabs in the previous version (V4) in a single view to provide an overview of changes without navigation.

Clicking a tab highlights it and the sheet's contents appear in the visualization panel (Figure 1). The purpose of this panel is to display the visual state of the tabs in different versions. Since our target users are Tableau specialists who are very familiar with the current interface of Tableau, but there are no APIs for extending Tableau workbooks, we decided to use screenshots of sheets and dashboards in our prototype to display the visualization panel. This panel enables easy transition back and forth to compare two versions of a tab. This approach was suggested by some of our study participants, *"for the visualization area maybe if I can swap before and after, then I can quickly look at the changes"*.

5.2.1. Change Matrix

In a preliminary study, we compared versions of two separate workbooks that were created collaboratively. We discovered that there was often a pattern in the changes created by an analyst at each revision. For example, if an analyst changes the aggregation function of an attribute from Sum to Average, then there is a high chance that she will repeat this change across most sheets in the workbook during that revision. As matrices are known to be effective visual representations for demonstrating patterns in data [Ber99, LPCC17], we used them to represent the change details.

Above the ribbon of tabs in the older version (V4 in Figure 1), we visualized the changes of each sheet within each of the change type categories presented in Table 1 (**DR2**). This *Change Matrix* shows the change types on rows and sheets in the older version (V4 in Figure 1) across the columns. Change types are ordered according to the significance of the changes as explained to us by the analysts. However, in our later evaluation study, we learned that some people may prefer an ordering that reflects the progression of the collaborative editing work, *e. g.,* "*There is a natural sequence of how these changes happen. [...] Typically, getting the data right happens first. Formatting happens last, [...]* " [P07].

In the Change Matrix, we represent the binary data of whether a specific change type was applied on a particular sheet with grey and white cells (**DR1**). For instance, Figure 1 shows that in transition

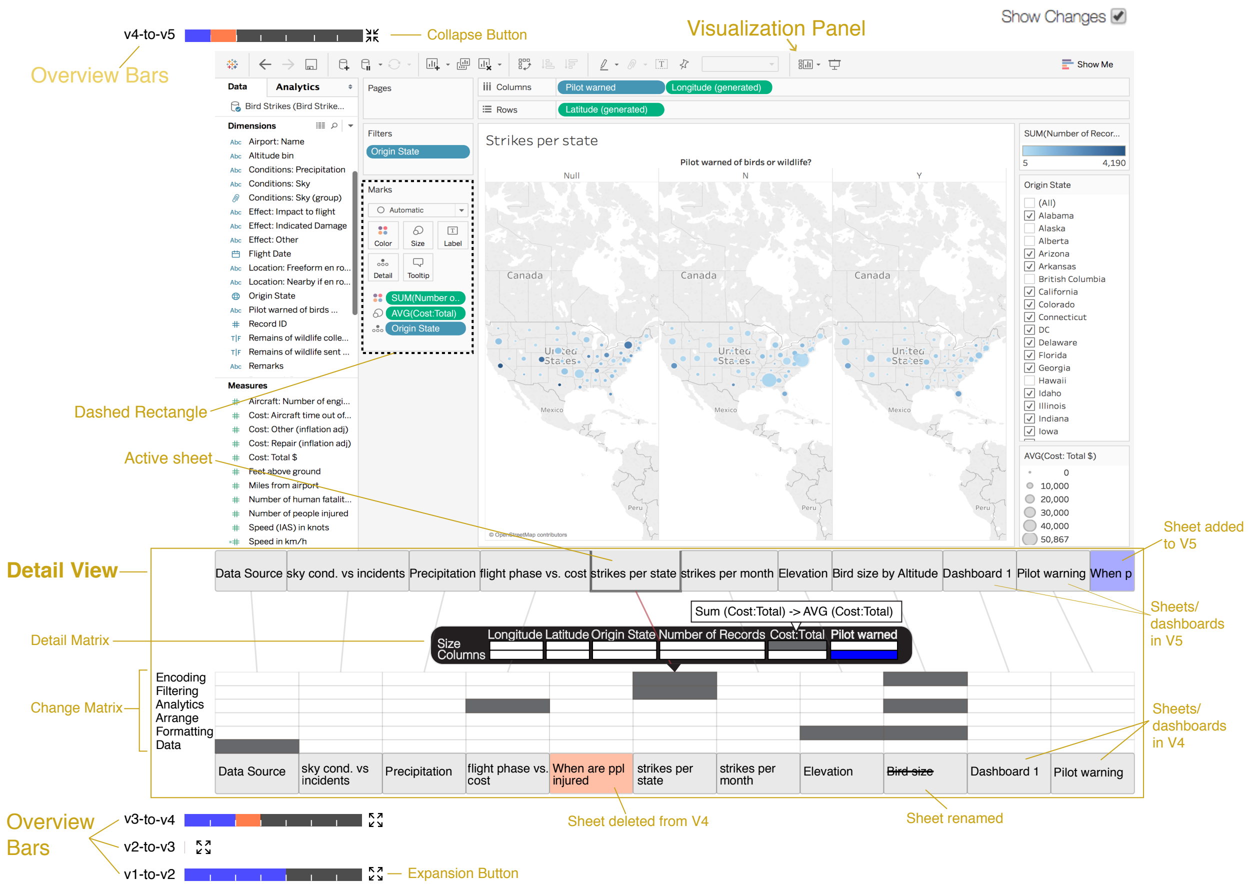

Figure 1: *Overview of ChangeCatcher consisting of Overview Bars, Visualization Panel, and Detail View. This example shows the differences between five consecutive versions of a Tableau workbook. The figure is annotated in yellow.*

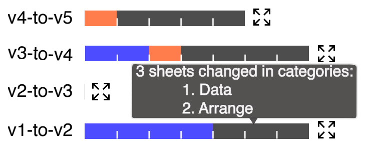

Figure 2: *Tooltips show detail about the underlying changes.*

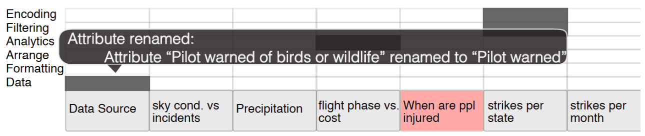

Figure 3: *A textual tooltip describing that an attribute was renamed.*

from V4 to V5, both "Encoding" and "Filtering" changes have been applied on the sheet "strikes per state". In an earlier paper prototype, we added 6 small squares to each sheet's tab (one per change category), colored by whether or not the sheet had a change of that type. This design took less space and was less visually busy than our current design, but made it difficult to identify the change type and to quickly scan all tabs for changes of a given type. In contrast, our matrix design enables visual scanning across the rows (to see where a particular change type happened) or the columns (to see what types of changes happened on a given sheet).

5.2.2. Detail Matrix

For drilling down into the detailed changes, we offer two levels of additional detail. Selecting a specific change type of a particular sheet (clicking its corresponding rectangle in the Change Matrix) 1) automatically navigates to bring a selected sheet into view, 2) shows the contents of the selected sheet in the Visualization Panel, and 3) shows further details in a small *Detail Matrix*.

We wanted a way to quickly reveal the detailed changes that was consistent with the rest of the design. Thus, we chose a small Detail Matrix to encode what changed vs. where it changed. The rows and columns contents of this matrix also depend on the represented change type. For instance, for the "Encoding" change type, the rows (Where) are the shelves in Tableau (*e. g.,* Rows, Columns, Color) that underwent a change, and the columns (What) are the data attributes used in the sheet (Figure 1). Similar to the color scheme we used in §5.1, a blue cell in the Detail Matrix represents a attribute's addition to a shelf, orange represents deletion, and grey indicates that the attribute has changed within the shelf. In Figure 1, opening the Encoding tooltip of sheet "strikes per state" reveals that attribute "Pilot warned" has been added to the shelf

"Columns", while attribute "Cost:Total" has been changed within the shelf "Size". By hovering the mouse over the grey cell: 1) a textual tooltip pops up containing the details of the change (*e. g.*, "Sum(Cost:Total) → Avg(Cost:Total)"), and 2) a dashed rectangle appears on top of the Visualization Panel to indicate the location where the change has taken place (see Figure 1). The position of these dashed rectangles in ChangeCatcher are calculated based on the current Tableau user interface and the location of its elements.

While this is a new visual representation, it has the advantage of giving a quick overview of the types of changes (*i. e.* deletion, addition, or change) without requiring people to read a list of changes to extract such information. Furthermore, our study showed that participants quickly learned how to read these visual matrices, and some even stated that they preferred reading through the matrix over reading textual descriptions, *e. g.*, "*As I get more experience, I can quickly scan these colors and get their immediate meanings. I feel tired reading all the text*" [P08]. As some of the changes could not be translated into the two dimensions of the matrix, we represent those changes as text. Figure 3 shows a textual tooltip to describe an attribute name change.

6. Usecase Scenario Walkthrough

Here we illustrate how ChangeCatcher can support a scenario identified in our requirements gathering. Suppose that Sarah is the head of the analytics department who checks workbooks before they are published. Alexa and Larry are creating a workbook about bird and aircraft collisions in the United States. Sarah reviewed the first version of the workbook and gave Alexa and Larry feedback. They created 4 more versions collaboratively while applying Sarah's feedback, and handed the final version off to Sarah to review.

Sarah opens up the latest version (V5) (Figure 4). She wants to identify the changes made by Alexa and Larry within the recent versions and determine whether the workbook needs to be revised again. In order to use ChangeCatcher to find the changes, Sarah checks the "*Show Changes*" checkbox on the top right. As a result, she observes a detail view of the most recent changes (V4-to-V5) as well as a high level overview of changes in the previous versions (Figure 5). The detail view between V4 and V5 shows that one sheet ("When are ppl injured") has been deleted (orange), one sheet has been added (blue), and one sheet has been renamed ("Bird size"→"Bird size by Altitude").

Looking at the Change Matrix, she observes all change types except Arrange. At this stage, Sarah is mainly interested in Formatting and Encoding changes. To examine them in more detail, she can click on the corresponding grey rectangles in the Change Matrix. For instance, clicking on the Encoding changes of the "strikes per state" sheet (see Figure 1) shows that "Pilot warned" has been added to the "Columns" (shown in blue), and "Cost:Total" has changed within the "Size" shelf (grey). Clicking back and forth on the "strikes per state" sheet in V4 and V5, Sarah can see the effects of the changes on the sheet. Clicking on the Formatting change of the sheet "Elevation" shows a tooltip stating that the title font size changed. Sarah also notices a Data change between V4 and V5. The tooltip reveals that "Pilot warned of birds or wildlife" is shortened to "Pilot warned" (Figure 3). Sarah thinks this is reasonable, so she does not need to discuss it with the developers.

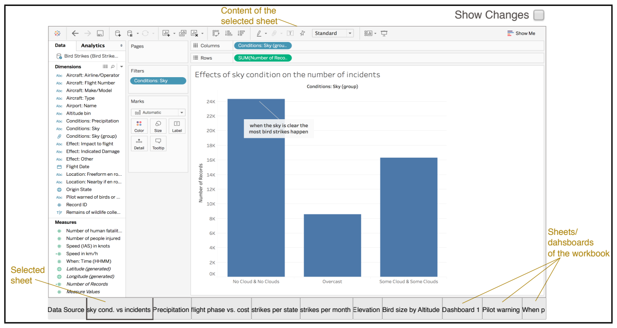

Figure 4: *A typical workbook without any changes shown.*

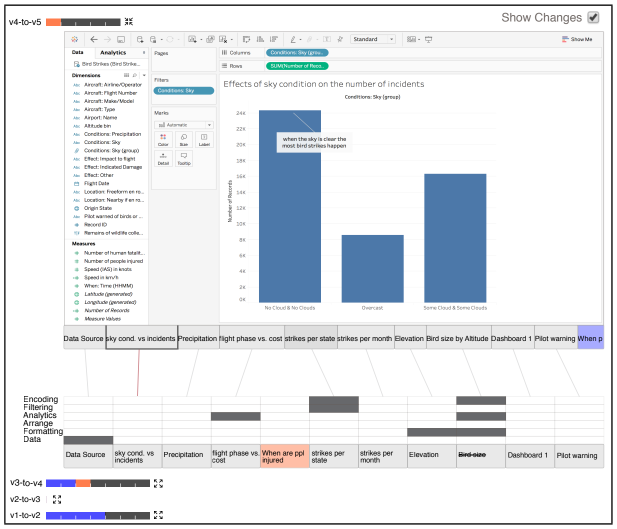

Figure 5: *Worbook status after checking the "Show Changes" checkbox. It shows the Overview Bar and Detail View of V4-to-V5 as well as Overview Bars of changes in previous versions.*

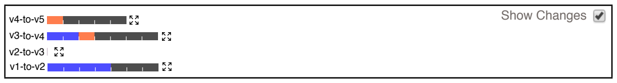

Figure 6: *Overview Bars of version changes with all details collapsed.*

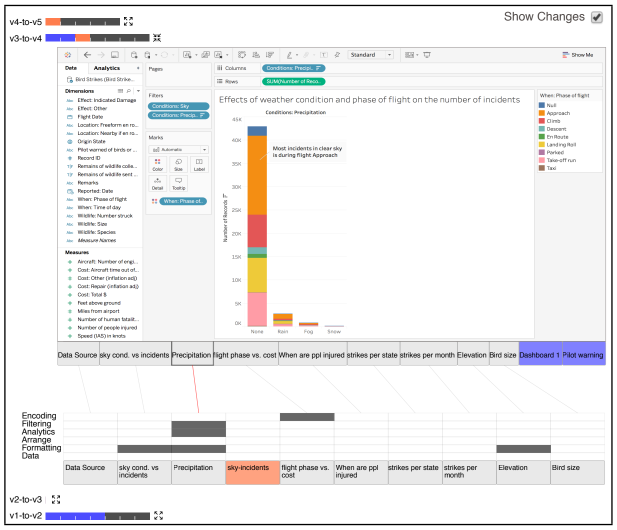

Figure 7: *ChangeCatcher showing the Detail View of V3-to-V4.*

After Sarah is finished examining V4-to-V5 changes, she closes the Detail View by pressing the collapse button (⤡). She then observes the Overview Bars of all versions (Figure 6). By hovering over the grey segment of V3-to-V4, Sarah finds that there are both Encoding and Formatting changes and she opens up the Detail View to examine them in more detail (Figure 7).

By examining the changes through ChangeCatcher, Sarah feels confident that she has not missed important changes as she writes her feedback for Larry and Alexa.

7. Experiential Response to ChangeCatcher

To gain feedback on ChangeCatcher's design ideas for visually representing version changes, we conducted 11 observational study sessions with Tableau professionals. We aimed to assess the value of its key features, gain feedback on the design, further develop guidelines for revealing visualization changes, and explore use cases for version difference tools in visualization. For ease of recruitment and analysis, participants took part as individuals rather than collaborating pairs. Our study simulated one step in a collaborative scenario where the participant receives an updated visualization from a fictional collaborator.

7.1. Method

Participants: We recruited 11 volunteers (5 female, 6 male, ages 18—65) from a software company. Each session lasted for 75 minutes. Tableau familiarity was a prerequisite, to ensure participants would have the necessary knowledge to understand Tableau workbooks and their changes. Participants came from a variety of backgrounds, including software engineering, user experience design, research, recruiting, sales, data analysis, and IT. Four participants stated that they needed to compare Tableau workbooks in their own work in either individual or collaborative scenarios. Participants took part in the study either in person (8 participants) or by videoconference (3 participants). Remote participants shared their screen so we could observe all interactions.

Procedure: We began with a brief training session, in which we introduced our change categories and walked participants through a sample scenario covering ChangeCatcher's features. Participants were given reference pages containing a description of the change types and potential task strategies. Following the training session, participants filled in a background demographic questionnaire.

We described a collaborative workbook authoring scenario, asking participants to imagine that they needed to review a collaborator's changes. Participants completed seven short-answer questions that required comparing versions of a workbook using the Bird Strike dataset [FAA]. Questions covered a variety of comparison cases (single worksheet / all sheets, overview / explore changes in depth, all change types / specific types). Participants then completed a full comparison of two versions of a workbook on Movie Success, identifying as many differences as possible within ten minutes.

Local participants worked through a paper task booklet and recorded their own answers. Remote participants worked through an electronic booklet and announced their answers aloud. The experimenter recorded the verbalized answers. Participants were asked to think-aloud while interacting with the prototype. Participants then completed a post-task questionnaire regarding the usefulness of system components and took part in a 30 minute semi-structured interview. Interviews focused on their experience completing the tasks and (when applicable) their own work tasks involving comparison of Tableau workbooks. Sessions were recorded using screen capture video plus audio.

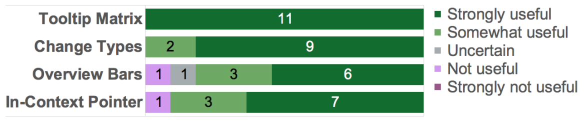

Figure 8: *Overall usefulness of features in ChangeCatcher.*

7.2. Questionnaire Results

Participants' ratings of the usefulness of various features are shown in Figure 8. All participants strongly agreed that the Tooltip Matrix provides useful information, *"...useful because I knew I wasn't missing anything"* [P01], and that the change types were useful, *"very useful because if the changes were just formatting and arrange, I say I don't need to care about these, but if it is encoding, or filtering or data, I may be like — 'oooh what did he do?'"* [P07].

However, there were mixed opinions about the Overview Bars. Nine participants found them to be strongly or somewhat useful, *e. g., "I like that there is a big picture and [...] lower level details"* [P06]. But two people felt the Change Matrix was sufficient, *"this [Change Matrix] is much more telling in terms of what has changed"* [P04]. Also, all but one participant found the dashed rectangles on the workbook useful, *"If I were not familiar with Tableau, like I don't know where you access a calculated field [...] that would really help me"* [P03], "*using this [dashed rectangle] as much as possible would be really helpful. It almost feels like you are interacting with your hands instead of your eyes. It has more of a feel to it*" [P07].

7.3. Interview Results

We manually transcribed and analyzed the data, grouping comments by emerging themes as in affinity diagrams [BH98]. Overall, participants were very positive about the approach and how they could use ChangeCatcher in their own work, *e. g.,* "*To me, this makes my day because I can be more self-service if someone reports a problem and then if I do have to interrupt someone [...] we can have a much more quick and detailed discussion*" [P07], "*[Let's say] Matt makes some changes right before he goes on vacation. This would be nice to see what he changed and say, 'oh now I see you made it so much more useful,' and we don't have to wait for him to get back*" [P09], "*this would probably make our day because we have an internal review process [...] would really help our team [...] quickly look together and say, 'let's see this calculation'*" [P02].

Participants reported that finding changes with ChangeCatcher was fast and straightforward. They also noted that there is no efficient way of doing such comparisons in Tableau, *"We don't have a great way of knowing what was changed. Was it a minor change, was it a major change, was the workbook completely overwritten or just tweaked the formatting. I don't want to pull out two copies of it, especially if it is a messy workbook"* [P19].

7.3.1. Increasing Confidence

ChangeCatcher helped users to feel confident that they had not missed any relevant changes, "*I have very high confidence that I understood all the changes that were listed*" [P04], "*I have been doing this task [in my own work] but it's weak right now. With this [ChangeCatcher] I have confidence that I know the scope*" [P11]. Moreover, improving confidence directly affects the time spent searching for changes. As ChangeCatcher visually displays the differences, people could be certain that they had seen them all.

7.3.2. Visually Assessing Work Progress

Participants reported that the overview bars and change categories might enable project managers to quickly assess overall work progress, "*I need a high level overview of changes. Maybe that's all I want to know. Just the numbers*" [P06], and "*my manager uses workbooks to track how much work we get done*" [P08]. Also, observing the change matrices across various versions and assessing which change types were applied at each step can summarize progress, "*I am early in the game, I expect to see data things [...]. But if I am late in the game I expect to see formatting, making sure it is ready for somebody to look at it*" [P05].

7.3.3. Factors Affecting the Importance of Change Types

Participants reported that change categories would be very useful because they give a quick overview of the changes they need to pay attention to, *e. g.,* "*it saves me a lot of time that you separated the changes and I can see the types of changes I might be interested in*" [P08], "*....very useful [...] depending on the type of collaboration you are doing, you can gloss over certain changes*" [P10]. In ChangeCatcher, we considered all the change types as equally important. However, our results revealed that the relative importance of the change types depends on many factors, described below.

Task characteristics: Participants' backgrounds and the types of tasks they performed influenced the types of changes that they considered important. For instance, those with data analytics and research backgrounds paid more attention to changes related to analytical work (*e. g.,* Data, Filtering, and Encoding), *"For me, some of the lightweight stuff like axis, font, and color formatting is not super important. But if I made an error in the calculated field or if I just did not aggregate a field correctly that can affect the analytical outcome, those are much more important"* [07]. In contrast, design experts paid more attention to the aesthetics including formatting and arrangement, *"When I create a vis, I do a lot of editing like the axis label or the title. For making it look good, you have to adjust a lot of formatting. "* [P03] and "*If I want the things pixel perfect, and had put a lot of thought into the format then I would be upset or wanting to know about all the different formatting changes that have been taking place so that I can verify them*" [P01].

Phase of development: Importance also depends on the phase of development, *e. g.,* "*If you are about to use this and publish it, everything matters, pixel by pixel. If you are building the dashboards and you are caring about the analytics, then you don't care about the formatting"* [P05] and *"Breaking the changes into categories give you a quick visual. Hey you are so stupid — why are you still changing the calculations? That's scary and hey no one should be changing those at this point"* [P04].

Expertise of editors: Knowing who has applied the changes may affect the review. If the editor is an expert in a specific area (*e. g.,* design), then collaborators may trust the changes without reviewing them. *"It depends on who made the changes. If they have a much better visual design sense than I do, then I might say fine, anything you do is probably better than what I would do."* [P05], *"if you trust their formatting ability, then ignore formatting changes."* [P10].

7.3.4. Supporting New Use Case Scenarios

Participants described several additional use case scenarios that ChangeCatcher could support, beyond our initial target scenarios:

Visual debugging and performance testing: Sarah is a test engineer whose workbook used to work; then, all of a sudden it takes too long to load. She needs to find all the differences between the working and broken versions to debug. *"When I am trying to track down a bug and I have made a change in the UI or a change in the workbook and wanna see if that affects anything, that's the time I really care about the differences"* [P01].

Self-service: Larry needs to make changes to a workbook that has been edited by an executive. However, he also wants to save the exec's time by not reviewing every single change with him. Thus, Larry first needs a self-service approach to find the changes by himself before going to the expert for explanation. *"People can self-service to find what change caused it before they reach out to the owner. This would really speed things up"* [P07].

Training: Jenny is a manager who wants to train her employees on certain analytic tasks. She creates a training workbook that captures her changes so that others can learn. *"I am trying to train my team. I don't want to put the solution in front of them"* [P09]. *"I see people make wonderful workbooks. I have not much idea how they made it. [...] We look at the logs to see how the workbook has changed. I think the matrix is a definite help"* [P06].

Remembering your own changes: Alicia decides to comment on some of her own changes so that her collaborators (or even herself) can remember the reasoning behind them. To do so, Alicia first needs to recall all the changes she made. Relying on memory for this task could be exhausting and error-prone. "*Right now we may hand write the changes in a notepad. The problem with a notepad is that people may forget things*" [P03], "*You change that tab instead of copying that and making a new one. And when you are done, you go and see oh what did I change. You know I don't want to do undo 16 times. I just want to remember what I did before*" [P10].

Division of work: Jack and Ross divide their authoring work into separate pieces, branching an original workbook. Before recombining their workbooks, Jack needs to make sure they have not changed the same sheet.

8. Re-examining the Challenges of Visualization Versioning

As we were developing ChangeCatcher and building our understanding of the real world challenges it could support, it became clear that comparing versions of visualizations is a unique task for which no good support tools exist. To gain a deeper understanding of current practice, we conducted another observational study with 11 new participants. While it is unusual to study a classical approach after developing a new system, we hoped that observing side-by-side workbook comparison first hand would enrich our understanding of *why* ChangeCatcher's design worked.

Participants had various job roles including engineering, user experience, and sales. All used Tableau regularly and four reported doing version comparison tasks in their own work. We employed the same task and protocol as in the study with ChangeCatcher, but focused on observing what best practices people would use if they only had access to the Tableau workbooks.

Many participants found the side-by-side comparison of workbooks exhausting, *e. g.,* "*I found it super manual. [. . .] If I had to be in this task all day, I'd probably hit my head against the wall*" [P02], and "*I am so glad I got a good sleep last night. It's quite taxing*" [P04]. Here we summarize key challenges participants encountered.

Mapping tabs between the versions: Tasks that required comparing all the tabs were the most time consuming. It was challenging to identify corresponding sheets, especially when order changed, sheets were added/deleted, sheets used default names (*e. g.,* Sheet1), or names changed, *"if things were completely renamed or deleted, I had to check and see if I missed a sheet"* [P01]. Participants also relied on the sheet order to find corresponding tabs and often assumed there was a deletion / addition if tab order changed. In contrast, ChangeCatcher participants did not encounter these problems.

Hidden meaning behind sheet order: Tab order often suggested the logical flow of the work, *"the tab order has a meaning. Like if you are following a framework [. . .] then the first tab is the first phase [. . .] When you change the orders, it might change the whole story"* [P05]. Sheet order could also indicate progress. *"I have a logical sequence of tabs in my mind [. . .] The draft tabs are at the beginning and the finished ones at the end [. . .] If my colleague moves [a tab] towards the end, it has the meaning to me that that's enough, don't work on that tab anymore"* [P05].

Recognizing subtle changes: Some changes were visible and obvious, "*some of these changes, like this annotation, I can pretty easily spot that on my own, [. . .] I don't need assistance identifying the change*" [P06]. However, participants needed help to find subtle changes, *"Changes in calculated fields are hard because those involve going into an editor, or a dialog, and changing it there. I would not notice if you change a formula"* [P10]. Changes to tooltips were also not immediately visible. "*By just going between sheets, I was worried that I did not catch something or something may have not changed between the screens but it has actually changed in the workbook*" [P07], or *"smaller visual clues are hard to catch"* [P05].

Comparing Tableau workbooks reminded participants of change tracking features such as those in Microsoft Word, Excel, or Acrobat, triggering some suggestions, *e. g., "I'd like to turn on a toggle button and then changes are laid on screen instead of looking at the previous and current version at the same time"* [P09]. Other features from these software tools, such as "Accept/Reject changes", were considered less useful, as a single change might have broad impact, *e. g., "Wouldn't need it [Accept/Reject] for Tableau because it's easy enough to make the change yourself with drag and drop. Plus, accept/reject individual features wouldn't make much sense because changes are tied to a whole series of other things"* [P11].

Overall, we observed that the time spent to find the changes was considerably higher than in the ChangeCatcher study. Average completion time per short-answer task was $131s$, compared to $32s$ for ChangeCatcher participants. While all tasks were more efficient with ChangeCatcher, this was less pronounced with more focused tasks. When the scope of the task problem was substantially narrowed, such as finding differences within a single sheet that had already been identified, side-by-side comparison was relatively feasible.

In contrast, participants took a very long time time to search for possible changes that did not exist, since this required an exhaustive search. Subtle changes such as small adjustments to fonts, changing an aggregation function, modifying the formula in a calculation, or changing the filtering of a data attribute had little visual impact on the workbook, making them difficult and time-consuming to notice. These issues were not a problem with ChangeCatcher.

9. Discussion and Future Work

9.1. Value of Revealing Visualization Workbook Differences

Our results clearly demonstrate the value of revealing changes between versions. In our study with ChangeCatcher, we observed that people felt confident that they had found all the changes, even when they were very subtle. ChangeCatcher specifically shines for revealing subtle changes due to its explicit representation of differences. In contrast, participants using Tableau workbooks had to look back and forth multiple times for changes that did not exist and were not confident that they had found all the changes. Increasing confidence can in turn 1) reduce time spent on discovering changes, 2) provide certainty regarding what changed, and 3) reassure people that their collaborators have not made unexpected changes. The only potential cost that participants mentioned was reduced screen space, " *You took my worksheet [. . .] and shrunk it down*" [P03]. This issue is mitigated by the fact that change information is exposed only on demand and can be hidden during normal editing.

9.2. Lessons Learned

The design requirements in section 4 were not immediately obvious from initial interviews. These insights were refined through iterative design and capture the most important lessons we learned. We rejected a design candidate that simply highlighted changes in the workbook itself, for failing to provide an overview of changes across all tabs. Likewise, we rejected a candidate that provided only an abstract overview of changes, for failing to explain the change in the context of the workbook itself. A third design that used indicators on each tab for different change types was rejected because it was difficult to quickly scan for all changes of a given type (e.g. formatting). In addition, we considered a side-by-side view of the old and new worksheet pages, but found that toggling back and forth both saved space and more successfully revealed changes that were not visually prominent: such changes pop-out visually when toggled. Thus, while our ultimate design may be simple, the insights of what information to reveal and at which level of detail were non-obvious. We hope these design insights prove helpful to designers of other versioning systems for visualization.

9.3. Study Limitations

Our studies are subject to some caveats. First, while we aimed to include a variety of participants who were all familiar with data analytics, only a few were professional data analysts who regularly needed to compare versions of workbooks. Thus, professional analysts, and particularly those who co-edit visualization workbooks, may have different behaviors and opinions than our participant group. Second, the workbooks in our study were unfamiliar to the participants. People's behavior may differ when they examine changes to their own workbooks, particularly workbooks with which they are intimately familiar. Third, our participants were Tableau experts with limited time availability. Thus, simulating a real collaboration among participants on a dataset that they were all interested to visualize and analyze was challenging. Therefore, we chose to simulate this collaboration through an imaginary scenario. Future longitudinal studies are needed to evaluate ChangeCatcher in the context of people's own collaborative scenarios.

9.4. Generalizability and Scalability

ChangeCatcher uses Tableau workbooks as an exemplar BI tool, but the general approach is not limited to Tableau. The core concept and design of identifying and representing change type per sheet can easily be applied to any tabular data visualization tool that uses a tab-based interface model to collect together multiple visualizations within a single file. With broader adaptation, it might be applicable to other tab-based data tools such as spreadsheets. Of course, some aspects of the design would need to be adapted to each tool, most notably the method of showing changes in the context of the interface (instantiated via dashed rectangles superimposed on the Tableau shelf interface in our implementation). In addition, while our change categories are broadly applicable to visualizations, a designer might want to tweak the categorization to match the interface or interaction model of a different tool. For example, our *analytics* category corresponded to features on Tableau's 'Analytics' tab and was named accordingly; this grouping made sense to our participants because of their familiarity with the Tableau interface.

Although, according to some participants, the number of changes between versions is typically small, "*I think most versions are going to be 20 to 30 changes*" [P04], scalability is another aspect to consider in future extensions. Our approach scales well to tens of sheets and data dimensions but may not scale well as the workbook size and the number of changes grow beyond that. Techniques such as filtering or collapsing sheets that are not the main focus could begin to address these scalability issues.

9.5. Extensions

We focused on comparing *consecutive* versions. The detail view could also directly compare *arbitrary* versions, assuming the system provided a mechanism to select specific versions.

Extending to real-time editing use cases (collaborative or individual) would require reconsidering the design (*e. g.,* one may need to see individual steps in addition to a summary of differences). ChangeCatcher's design explicitly hides individual change steps in favor of a clear overview, precisely the level of detail needed for change review tasks. Yet participants pointed out that for a few tasks, being able to access the order of changes could also be useful, "*I can see that they have made the changes. Part of understanding the changes for me is to understand the order that they did things. Like maybe they added a calculated field at the very beginning and that's interesting to know versus adding that at the very end*" [P01], *"The sequence of changes might give me a look at how they were thinking and give me an idea of whether their changes make sense"* [P08]. Revealing the change order might also be useful for training tasks. Adding a playback feature to ChangeCatcher might be one way to reveal change order on demand. A related extension could enable sheets to be reverted to their previous state, valuable for undo tasks.

We focused on *identifying* version changes. Providing the ability to record the *reasoning* behind the changes and mark high priority items for review by others was also requested, *"...what that person was thinking when they were making the changes, because I can see that they have made the changes but I don't know why"* [P01]. Integrating such commentary would be a useful topic for future work, but does not negate the need to first identify what has changed.

Another challenge is to assess and reveal the *impact* of a change, "*It's difficult to quantify how impactful a change was. Because you could have a very impactful change that only affected the data source but that can cascade across the entire workbook*" [P02]. Data source and calculation changes, in particular, could potentially impact every sheet in the workbook or might have no impact at all. Similarly, we observed in our own work with Tableau that related changes may be applied repeatedly (*e. g.,* an editing pass to ensure title consistency). Here ChangeCatcher would show many changes that may be conceptually easier to think of as a group. Exploring ways to group related changes and quantify their impact would thus be valuable future work.

10. Conclusion

We have taken a close look at tasks that involve understanding changes between versions of visualizations. Through our initial discussions with experts, we found that this is an important and difficult task particularly in asynchronous collaboration, but also in some types of individual work. We describe design requirements and several different use case scenarios, which may prove useful in designing future version difference visualizations.

With ChangeCatcher, we introduced a technique that enables people to visually compare versions of a visualization workbook, illustrating potential design approaches for revealing version changes. We describe design considerations for version difference tools, including required functionality and a categorization of change types that our participants found useful in itself. Participants in our study reported that ChangeCatcher's design facilitated comparison tasks by providing an overview that could be used to rapidly locate changes by type or by location.

This work demonstrates the value of explicitly revealing version changes to support collaborative editing of visualizations and introduces a technique to visually reveal differences. As our work on developing visual support for version comparison progressed, we became more aware that just like versioning needs are different for document authoring and for software engineering, visualization versioning requires its own specialized support. We hope that our work lays the foundation for further research into a new and important aspect of visualization and visual analysis.

Acknowledgments

We thank Heidi Lam for design suggestions and help with log files.

References

[AKP14] Abraham J., Kannampallil T., Patel V. L.: A systematic review of the literature on the evaluation of handoff tools: implications for research and practice. *Journal of the American Medical Informatics Association 21*, 1 (2014), 154–162. 1, 2

[Aus05] Australian Commission on Safety and Quality in Healthcare: *Clinical Handover and Patient Safety Literature Review*, March 2005. 2

[Ber99] Bertin J.: Readings in information visualization. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999, ch. Graphics and Graphic Information Processing, pp. 62–65. 4

[BH98] Beyer H., Holtzblatt K.: *Contextual Design*. Morgan Kaufmann Publisher, Inc., San Francisco, 1998. 7

[BM13] Brehmer M., Munzner T.: A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (Dec 2013), 2376–2385. 3

[BOH11] Bostock M., Ogievetsky V., Heer J.: D^3 data-driven documents. *IEEE transactions on visualization and computer graphics 17*, 12 (2011), 2301–2309. 4

[EGK*01] Eick S. G., Graves T. L., Karr A. F., Marron J. S., Mockus A.: Does code decay? assessing the evidence from change management data. *IEEE Transactions on Software Engineering 27*, 1 (Jan 2001), 1–12. 2

[ESS92] Eick S. C., Steffen J. L., Sumner E. E.: Seesoft-a tool for visualizing line oriented software statistics. *IEEE Transactions on Software Engineering 18*, 11 (Nov 1992), 957–968. 2

[FAA] FAA: Aircraft wildlife strike data. URL: https://catalog.data.gov/dataset/aircraft-wildlife-strike-data-raw-data-382d6. 7

[FSC*06] Freire J., Silva C. T., Callahan S. P., Santos E., Scheidegger C. E., Vo H. T.: Managing rapidly-evolving scientific workflows. In *International Provenance and Annotation Workshop* (2006), Springer, pp. 10–18. 2

[FSJC13] Forbes A., Surdeanu M., Jansen P., Carrington J.: Transmitting narrative: An interactive shift-summarization tool for improving nurse communication. In *Proceedings of the IEEE Workshop on Interactive Visual Text Analytics (TextVis)* (2013). 2

[GAW*11] Gleicher M., Albers D., Walker R., Jusufi I., Hansen C. D., Roberts J. C.: Visual comparison for information visualization. *Information Visualization 10*, 4 (2011), 289–309. 2

[HFR*10] Hartmann B., Follmer S., Ricciardi A., Cardenas T., Klemmer S. R.: D. note: revising user interfaces through change tracking, annotations, and alternatives. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2010), ACM, pp. 493–502. 2

[HM76] Hunt J. W., MacIlroy M.: *An algorithm for differential file comparison*. Bell Laboratories New Jersey, 1976. 1, 2

[HMSA08] Heer J., Mackinlay J., Stolte C., Agrawala M.: Graphical histories for visualization: Supporting analysis, communication, and evaluation. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (Nov 2008), 1189–1196. 2, 3

[HVHC*08] Heer J., Van Ham F., Carpendale S., Weaver C., Isenberg P.: Creation and collaboration: Engaging new audiences for information visualization. In *Information Visualization*. Springer, 2008, pp. 92–133. 2

[LB85] Lehman M. M., Belady L. A. (Eds.): *Program Evolution: Processes of Software Change*. Academic Press Professional, Inc., San Diego, CA, USA, 1985. 2

[LM12] Loeliger J., McCullough M.: *Version Control with Git: Powerful tools and techniques for collaborative software development*. O'Reilly Media, Inc., 2012. 1

[LPCC17] Loorak M. H., Perin C., Collins C., Carpendale S.: Exploring the possibilities of embedding heterogeneous data attributes in familiar visualizations. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (jan 2017), 581–590. 4

[LWPL11] Lu J., Wen Z., Pan S., Lai J.: Analytic trails: supporting provenance, collaboration, and reuse for visual data analysis by business users. In *IFIP Conference on Human-Computer Interaction* (2011), Springer, pp. 256–273. 2

[MGH05] Mehra A., Grundy J., Hosking J.: A generic approach to supporting diagram differencing and merging for collaborative design. In *Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering* (2005), ACM, pp. 204–213. 2

[MIC] MICROSOFT CORPORATION: Power bi. URL: https://powerbi.microsoft.com/. 1

[Mun14] Munzner T.: Marks and channels. In *Visualization Analysis and Design*. CRC Press, 2014, ch. 5, pp. 94–114. 4

[NCK*92] Neuwirth C. M., Chandhok R., Kaufer D. S., Erion P., Morris J., Miller D.: Flexible diff-ing in a collaborative writing system. In *Proceedings of the 1992 ACM Conference on Computer-supported Cooperative Work* (New York, NY, USA, 1992), CSCW '92, ACM, pp. 147–154. 1

[PA06] Petschnigg S. M. D. G., Agrawala M.: Comparing and managing multiple versions of slide presentations. In *UIST: Proceedings of the ACM Symposium on User Interface Software and Technology* (2006), Association for Computing Machinery, pp. 47–56. 2

[PM09] Paul S. A., Morris M. R.: Cosense: enhancing sensemaking for collaborative web search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2009), ACM, pp. 1771–1780. 2

[QLI] QLIK:. URL: https://www.qlik.com/. 1

[RESC16] Ragan E. D., Endert A., Sanyal J., Chen J.: Characterizing provenance in visualization and data analysis: an organizational framework of provenance types and purposes. *IEEE transactions on visualization and computer graphics 22*, 1 (2016), 31–40. 2

[Sha08] Sharma N.: Sensemaking handoff: When and how? *Proceedings of the American Society for Information Science and Technology 45*, 1 (2008), 1–12. 2

[ST15] Sarvghad A., Tory M.: Exploiting analysis history to support collaborative data analysis. In *Proceedings of the 41st Graphics Interface Conference* (2015), Canadian Information Processing Society, pp. 123–130. 2

[SvW08] Shrinivasan Y. B., van Wijk J. J.: Supporting the analytical reasoning process in information visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2008), CHI '08, ACM, pp. 1237–1246. 2

[TAB] TABLEAU SOFTWARE:. URL: https://www.tableau.com/. 1

[Tic85] Tichy W. F.: Rcs—a system for version control. *Software: Practice and Experience 15*, 7 (1985), 637–654. 1

[TLZD16] Tharatipyakul A., Lee H., Zhao S., Davis R. C.: Supporting the comparison of alternative stories. In *Proceedings of the 28th Australian Conference on Computer-Human Interaction* (New York, NY, USA, 2016), OzCHI '16, ACM, pp. 266–270. 2

[Tuf91] Tufte E. R.: Envisioning information. *Optometry & Vision Science 68*, 4 (1991), 322–324. 2

[VTvW05] Voinea L., Telea A., van Wijk J. J.: Cvsscan: Visualization of code evolution. In *Proceedings of the 2005 ACM Symposium on Software Visualization* (New York, NY, USA, 2005), SoftVis '05, ACM, pp. 47–56. 2

[WDC03] Wang Y., DeWitt D. J., Cai J. Y.: X-diff: an effective change detection algorithm for xml documents. In *Proceedings 19th International Conference on Data Engineering (Cat. No.03CH37405)* (March 2003), pp. 519–530. 2

[WHHA11] WILLETT W., HEER J., HELLERSTEIN J., AGRAWALA M.: Commentspace: structured support for collaborative visual analysis. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (2011), ACM, pp. 3131–3140. 2

[XAJK*15] XU K., ATTFIELD S., JANKUN-KELLY T., WHEAT A., NGUYEN P. H., SELVARAJ N.: Analytic provenance for sensemaking: A research agenda. *IEEE Computer Graphics and Applications 35*, 3 (2015), 56–64. 2

[XS05] XING Z., STROULIA E.: Umldiff: An algorithm for object-oriented design differencing. In *Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering* (New York, NY, USA, 2005), ASE '05, ACM, pp. 54–65. 2

[ZGI*18] ZHAO J., GLUECK M., ISENBERG P., CHEVALIER F., KHAN A.: Supporting handoff in asynchronous collaborative sensemaking using knowledge-transfer graphs. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (Jan 2018), 340–350. 1