




Seamless Collaborative Coding with Visualization

F. Becker¹  R. P. Warnking²  and T. Blascheck¹ 

¹University of Stuttgart, Institute for Visualization and Interactive Systems, Germany

²Heidelberg University, Medical Faculty Mannheim, Department of Biomedical Informatics, Germany

Abstract

Coding data is a time-consuming activity, but one that is found in many places like scientific research and public online spaces. We developed the CollaCode system to support collaborative coding for a video game dataset, leveraging visualization with ubiquitous editing for seamless transitions between editing data and analyzing coding results. CollaCode employs multiple visualizations to support different activities in the coding workflow: tagging data, analyzing coding results, and resolving disagreements. Compared to existing approaches, our system explicitly models coding iterations, allowing coders to understand tag provenance while giving more transparency to the coding process. We discuss challenges in collaborative coding settings, how our system design addresses these challenges, and opportunities we see for future work.

CCS Concepts

• **Human-centered computing** → **Visualization systems and tools; Collaborative and social computing systems and tools;**

1. Introduction

Coding data is a fundamental activity in many domains that comes in different forms. Wikis employ social tagging [GLYH10] to organize knowledge, whereas researchers use methods like thematic analysis [Boy98] or grounded theory [GSS68] for qualitative data analysis (QDA). Each tagging scenario comes with its own context-dependent requirements. Researchers often code text from studies and interviews, whereas social tagging looks at data more holistically. Existing systems for QDA, like Atlas.ti or MAXQDA, are not designed to handle abstract data that cannot be split into meaningful parts. They assume strictly separated workflow stages and scale poorly for larger tag hierarchies. Born from the needs of a nine-months long coding project investigating 388 video games [BWBB25], we iteratively developed the CollaCode system to suit our project's needs. CollaCode treats data as more abstract, it allows coders to find and resolve disagreements, and records tag provenance to make the iterative nature of coding visible. Visualizations combined with *ubiquitous editing* enable seamless transitions between tagging and analysis to mirror our experience of how different coding activities interleave. Code for the CollaCode system is available at <https://github.com/ArielMant0/collacode>.

2. Related Work

Motivations and requirements for digital coding systems vary by use case. In social tagging, motivations range from sharing knowledge and self-presentation to facilitating information retrieval [GLYH10]. Systems that support data annotation to curate labels for machine learning [FDB18, FWZ*20, XYX*19, ZNX*23]

focus on optimizing the trade-off between label quality and required human effort. Labeling is often a less creative activity than coding, since labels are not derived from data, but it still requires human involvement and collaboration [MWA*21]. Researchers engaging in QDA are more interested in finding insights grounded in data. Existing systems like Atlas.ti (atlasti.com), MAXQDA (maxqda.com), or Condens (condens.io) enable multiple coders to create and analyze codings for text, image, or video data. None of these systems allow coders to tag artifacts as a whole, some only allow shallow tag hierarchies (2-3 levels), real-time collaboration is usually not possible, only a few support analysis of tagging disagreements, and none track changes over time. Combining these shortcomings with a steep learning curve, many researchers have been found to only use these systems for data management and initial coding phases, if at all. [JWFB21, KR24]

Other research related to coding investigates how machine learning can be integrated into the coding process, focusing on suggesting tags for new data [OSDR24, RM21], finding disagreement [GCC*24, GGL*24], or clustering documents [HMF*22]—always working with text data. In the area of visualization, Blascheck et al. [BBB*16] developed a visual analytics system that integrates rich study data, including transcripts, videos, eye tracking data, and interaction logs for analysis and coding. They use word-sized visualizations to support multiple comparison tasks but do not let the coder analyze disagreements or coding provenance. Aoenium [DCS*17] employs machine learning to highlight disagreements, but the system does not generalize to other usage scenarios than text, supports only two coders, and its components are not designed for tag collections larger than 10 or 12 because

tags are chosen from a text list and are encoded through color in different visualizations. CodeWizard [GOM18] is embedded in Microsoft Excel and designed to support collaborative coding with a low learning barrier while helping coders find sources of disagreement. The authors use matrix visualizations to display correlated disagreement, but their system does not support hierarchical tag structures, it requires manual labor to combine data from different coders, and it is not designed to support larger tag collections.

3. Designing for Collaborative Coding

In this section, we explore challenges for collaborative coding and how we designed CollaCode to address them. Regardless of the specific context, challenges for collaborative coding include the coordination of collaborative work, the development of satisfactory codes through data analysis, and the communication of results to others. [KR24] QDA in particular is a labor-intensive and creative process where ideas and data are constantly re-examined and adjusted. It's a process of constant reflection, where both data and codes are analyzed at different scales. [Bas03] Collaboratively creating and evaluating codes means being able to code individually, but also share codes, find and understand the differences, and implement changes with little effort. Provenance is an integral part of this process, helping coders to understand how they arrived at specific insights and how codes changed over time. It can also support communication with stakeholders, especially when there are differences in experience regarding qualitative methods. [KR24]

A challenge specific to our context was the unconventional nature of our data: video games. While single aspects of a video game can be represented, like the design of a menu or scene, capturing every aspect is difficult—even in rich data like hours-long video footage. As a consequence, our system needed to handle data in a more abstract manner. Instead of tagging smaller parts like sentences in a document, one artifact is assigned a collection of tags. To support reasoning about assigned tags in the absence of rich source data, we include *evidence*, which associates additional data (text, images, or video) with an artifact-tag tuple.

3.1. CollaCode Design

The CollaCode system was developed to address the previously outlined challenges for our coding scenario to different degrees, also keeping scalability [RPA*24] in mind to support diverse datasets. Primarily, it supports coders during the coding process on an individual and collaborative level. That means coders must be able to easily enter or manipulate data (C_1), build understanding of their codes and patterns therein (C_2), and compare codes between coders and resolve disagreements (C_3). To tackle C_1 , CollaCode implements the concept of *ubiquitous editing*. In systems that support numerous or complex interactions, these are sometimes reserved to specific locations of the system. To perform a specific action, users must navigate to another location, perform the action, and then navigate back. In the worst case, they even have to reconstruct their previous application state. CollaCode aims to minimize such disruptions through *ubiquitous editing*. Any entity representation—be it an artifact, a tag or a piece of evidence—lets coders initiate actions via right-clicking, without leaving the cur-

rent context. This allows for *seamless* transitions between different coding activities that we experienced as highly interleaved, like coding an artifact, then reflecting about the usage of a tag, consulting other artifacts with the same tag, and finally making a coding decision. To support C_2 , CollaCode includes summary visualizations that give coders overviews and support *ubiquitous editing* as well as filtering. We also directly model the iterative nature inherent to many coding approaches via tag provenance. At any point, coders can create a transition, which creates a copy of the current coding and creates links between copied tags. This allows coders to understand changes between coding iterations, available through visualizations in CollaCode. For successful coordination of collaborative work (C_3), CollaCode lets coders directly work together on one coding. Coders can choose whether they want to see only their own or everyone's tag assignments, which is respected by all components and visualizations of tags. Thereby, each coder can add assignments without undue priming, but can also review and discuss the current coding state on a shared platform. To help coders find instances with disagreement, we calculate Krippendorff's alpha [Kri19] for tags and artifacts. Disagreements can be resolved directly, either in-bulk via *ubiquitous editing* or by specifying the desired constellation of tag assignments. CollaCode additionally provides dedicated views that focus on data exploration, which can be used to share results with others. These views also employ visualizations that summarize data, like showing the tag co-occurrences in a matrix or displaying the distribution of evidence over tags.

3.2. System Components

CollaCode is broadly structured into views based on different coding activities, but it reuses the same types of components and visualizations to reduce overall complexity. Visualizations in CollaCode serve three different functions: (i) they provide an overview of specific data such as tag frequencies, (ii) they allow coders to define filters, and (iii) they can be used to edit or add data such as tag assignments or evidence. To support both smaller and larger tag hierarchies, we opted for dense pixel-oriented visualizations [Kei00] that can show different attributes in relation to the hierarchical structure. In the following, we describe the views in CollaCode that focus on coding activities and how they implement previously described design ideas. Extended descriptions and examples for additional system views are available in the supplemental material.

3.2.1. Coding View

In the coding view, coders can see a tag summary and an interactive table of all artifacts. The tag summary combines a node-link diagram with a colored bar-code to visualize tag frequencies in a space-efficient manner, similar to Figure 1 a & b. The table lists all artifacts and their associated tags, which can be displayed as text or as a colored bar-code. The bar-code visualizes which tags a coder has assigned to an artifact, and clicking any bar toggles the respective tag assignment, letting coders change data on the fly. Clicking on a row in the table opens the artifact editor, which allows coders to modify tag assignments and evidence. It employs a collapsible treemap with uniform node weights to visualize the tag hierarchy. Coders can simply click on any node to toggle its tag assignment, while still having an overview of the complete tag hierarchy. A tag's

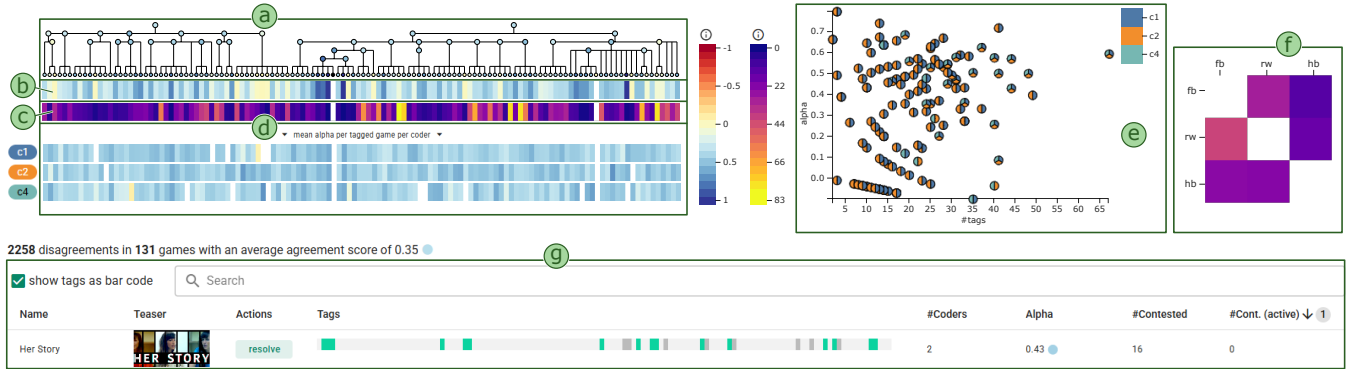


Figure 1: The agreement view that supports exploration of inter-coder agreement using Krippendorff's alpha [Kri19]: (a) node-link diagram depicting the tag hierarchy showing mean alpha values for intermediate nodes, (b) bar-code visualizing alpha values per tag, (c) bar-code visualizing tag occurrences, (d) bar-codes visualizing mean alpha values per coder, (e) scatter plot showing the number of tags compared to alpha values for each artifact, (f) matrix showing when coders assigned the same tag, and (g) top row of the artifact table showing which tags are assigned (grey) and for which there is disagreement (teal).

associated evidence is displayed as a small dot in the respective tree node, allowing coders to view or edit it. Both the evidence dots and all treemap nodes support *ubiquitous editing*.

3.2.2. Agreement View

In the agreement view, coders again see a tag summary, this time showing Krippendorff's alpha for each tag (cf. Figure 1 a & b) in addition to tag occurrences (cf. Figure 1 c). Krippendorff's alpha is a general measure that works for any number of coders, handles any type of data, and allows for missing values. It gives coders an idea of how well they agree on the use of a specific tag or indicates the overall agreement for a single artifact over all tags that coders assigned. An additional color-coded bar-code for each coder visualizes the mean alpha values for all artifacts the coder has assigned a specific tag to (cf. Figure 1 d). Next to these bar-codes, we show the relation between the number of tags and alpha values for artifacts in a scatter plot (cf. Figure 1 e). Each artifact in the scatter plot is visualized as a pie glyph, encoding which coders have tagged the artifact via color. A small color-coded matrix displays how often pairs of coders tag together (cf. Figure 1 f). All components support filtering in various ways like clicking on bars, clicking on legend labels, or brushing in the scatter plot. Below these visualizations, the agreement view includes a modified artifact table listing artifacts (cf. Figure 1 g). It includes the alpha value of an artifact and its assigned tags. In its default configuration, coders can see all tags assigned to an artifact as text with small colored ovals indicating who assigned the tag. Tags with disagreement are highlighted by rendering all agreed upon tags with lower opacity. Akin to the coding view, coders can choose between text and the bar-code to represent tags. For the latter, the color encoding is binary, indicating tag (dis-)agreement.

Going beyond finding disagreement, CollaCode gives coders the means to resolve their disagreements. An interactive table lets coders choose which tags each of them want to assign. Each row represents a tag and displays the tag name, related evidence, and a cell for each participating coder. Clicking on a coder's cell changes

the tag assignment for that coder. The background color of a cell indicates whether a tag was initially assigned, letting coders compare the current to the initial tag constellation. Using this table, it only takes a few clicks to add or delete many tag assignments for different coders. Both tag and evidence representations support ubiquitous editing, letting coders edit tag descriptions, add new evidence, or browse artifacts for a tag.

3.3. Provenance View

In other approaches, multiple iterations of codes are not modeled explicitly, even though the iterative process of coding is often emphasized as a crucial element to achieving good results. In the provenance view, CollaCode lets coders view how the tag hierarchy has changed between iterations. Reusing the design of the bar-code visualizations, it shows tag provenance for pairs of codes. Tags are sorted on the horizontal axis by their position in the tree and displayed as bars. Bar height encodes depth in the tree, and color encodes the number of tag occurrences. The preceding code is shown on the top (cf. Figure 2 a), while the succeeding code is placed at the bottom (cf. Figure 2 b). Vertical links connect tags that are tracked from one code to the other, and coders can switch between different filters to highlight tags that were removed, added, split, or merged. Each bar supports ubiquitous editing and creating filters. When tags are selected, an action panel (cf. Figure 2 c) enables coders to perform advanced modifications like splitting or merging tags. To make these complex actions easier to understand, coders get a preview of what the action will do before performing it.

4. Discussion

The CollaCode system was born from the needs of coding 388 video games—rich data that is difficult to represent completely. We needed a system that could handle such abstract data, enable direct collaboration for three coders, support *working with* and *analyzing* larger datasets, and track changes over time. While other approaches fulfill some of these requirements, none of them manage

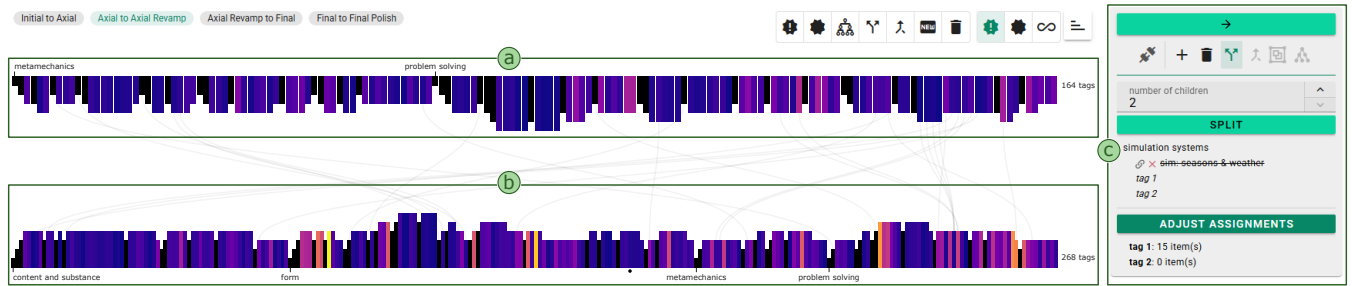


Figure 2: Provenance visualization showing changes from one coding iteration (a) to the next (b). Each bar represents a tag, with height encoding depth in the hierarchy and color encoding the number of occurrences. A link between two tags indicates a connection between iterations. The action panel (c) lets coders perform advanced actions like splitting tags, changing tag connections, and more.

to fulfill all of them adequately. In the following, we discuss design decision, insights from the design process, and system limitations.

CollaCode takes advantage of pixel-oriented bar-code visualizations that are space-efficient which act as overviews for different attributes in different parts of the system. Although space-efficient, large hierarchies with thousands of tags are too large for CollaCode to visualize effectively. Interacting with small elements in these visualizations can also be difficult for some users, which can lead to frustration or mistakes. To limit the overall system complexity, we reused the same bar-code design to reduce learning barriers and added shortcuts to allow quick switching between data analysis and data editing. Coding is a time-consuming activity shaped by individual perspectives and constant reflection, with new ideas affecting existing ones and vice versa. A vital design choice that solidified during our coding project was to minimize disruptions from context changes and simplify data modification. Every data representation is *functional*, acting as a shortcut to modification and elaboration. In our project, that meant quickly adding associated evidence, viewing other coded examples, changing tag assignments, or resolving disagreements in bulk. Other scenarios and users may have different needs—designing good coding systems requires understanding the different mental and tangible steps coders take along the way. Besides making transitions between steps *seamless*, direct manipulation [Shn83] in visualizations can make it easier to edit data. CollaCode includes this only in a basic version (e.g., changing tag assignments in a treemap), but letting coders easily perform large data edits could help in making complex codes more viable to construct. Although related works are looking towards machine learning and language models for help, these may take away what coders value in QDA. [MT18, JWFB21] Consequently, we should also consider how to design systems that support coders in creatively working *with* their data.

Whilst CollaCode is primarily designed to support coders during and after coding, it also contains views that stakeholders can use to explore the data. Since it is a web-based system and does not require an account to view, it can easily be shared with others. These views and visualizations may not be useful for every stakeholder, especially those not at all familiar with the data, but they can still help coders in explaining their results. Reporting results can be just as important as coding itself, if not more so. [KR24] Our system does not include any type of report generation, but contains

the data foundation to do so—especially with the existing provenance data. Although CollaCode has already been used in a scientific project in which three coders collaboratively coded video games over nine months, it has not yet been evaluated formally. As a proof-of-concept, we imported data for two usage scenarios into the system to explore how these would work in CollaCode. The details of these scenarios and the data processing are described in the supplemental material. Effectively evaluating a coding system can be difficult since coding requires a lot of time and coders are reluctant to spend this time learning or experimenting with a new system. Evaluating usability can be informative, but may not be representative of the value a system brings to the coding process. Yet, evaluations are necessary to understand how these design choices support or hinder collaborative coding.

5. Conclusion

With CollaCode, we present a collaborative coding system that addresses different challenges for collaborative coding. It was developed iteratively during a coding project analyzing video games with three coders over nine months, but was not evaluated systematically. CollaCode considers scalability in its design by using space-efficient visualizations and it reduces disruptions by making transitions between analysis and data modification seamless. The latter proved a frequent need during our own coding process, which CollaCode achieves by combining visualizations with ubiquitous editing. Collaboration is a crucial part of coding and is also included in CollaCode's design. Coders can choose whether they want to see other coder's tag assignments, share a global tag hierarchy, and are supported in both finding and resolving disagreements. In contrast to existing solutions, our system explicitly models coding iterations to enable provenance analysis for tags and increase coding transparency. However, there are more opportunities to record and visualize provenance data. For example, per-artifact provenance could allow coders to understand changes made to a single artifact over all coding iterations. Another problem we faced during coding was the act of *coordinating* collaboration. Making notes during coding, documenting collaborative decisions, or contesting other coders' tag assignments all took place outside of our system. This requires considerable effort that could be reduced by integrating such features for collaborative decision making and decision provenance into the system itself.

Acknowledgments

Tanja Blascheck is funded by the European Social Fund and the Ministry of Science, Research and Arts Baden-Württemberg. We want to thank Hendrik Brückler for his continued feedback and support in designing the CollaCode system.

References

- [Bas03] BASIT T.: Manual or electronic? the role of coding in qualitative data analysis. *Educational Research* 45, 2 (2003), 143–154. doi:10.1080/0013188032000133548. 2
- [BBB*16] BLASCHECK T., BECK F., BALTES S., ERTL T., WEISKOPF D.: Visual analysis and coding of data-rich user behavior. In *2016 IEEE Conf. Vis. Analyt. Sci. Tech. (VAST)* (2016), IEEE, pp. 141–150. doi:10.1109/VAST.2016.7883520. 1
- [Boy98] BOYATZIS R. E.: *Transforming qualitative information: Thematic analysis and code development*. SAGE, Thousand Oaks, Calif., 1998. 1
- [BWBB25] BECKER F., WARNKING R. P., BRÜCKLER H., BLASCHECK T.: Beyond Entertainment: An Investigation of Externalization Design in Video Games. *Computer Graphics Forum* (2025). 1
- [DCS*17] DROUHARD M., CHEN N.-C., SUH J., KOCIELNIK R., PENA-ARAYA V., CEN K., ZHENG X., ARAGON C. R.: Aeonium: Visual analytics to support collaborative qualitative coding. In *2017 IEEE Pacific Visualization Symposium (PacificVis)* (Piscataway, NJ, 2017), Weiskopf D., Wu Y., Dwyer T., (Eds.), IEEE, pp. 220–229. doi:10.1109/PACIFICVIS.2017.8031598. 1
- [FDB18] FELIX C., DASGUPTA A., BERTINI E.: The exploratory labeling assistant. In *Proc. 31st Annual ACM Symposium on User Interface Software and Technology* (New York, NY, 2018), Baudisch P., (Ed.), ACM Conferences, ACM, pp. 153–164. doi:10.1145/3242587.3242596. 1
- [FWZ*20] FAN M., WU K., ZHAO J., LI Y., WEI W., TRUONG K. N.: Vista: Integrating machine intelligence with visualization to support the investigation of think-aloud sessions. *IEEE Trans. Vis. Comp. Graph.* 26, 1 (2020), 343–352. doi:10.1109/TVCG.2019.2934797. 1
- [GCC*24] GAO J., CHOO K. T. W., CAO J., LEE R. K.-W., PERRAULT S.: Coaicoder: Examining the effectiveness of ai-assisted human-to-human collaboration in qualitative analysis. *ACM Trans. Comp.-Hum. Inter.* 31, 1 (2024), 1–38. doi:10.1145/3617362. 1
- [GGL*24] GAO J., GUO Y., LIM G., ZHANG T., ZHANG Z., LI T. J.-J., PERRAULT S. T.: Collabcoder: A lower-barrier, rigorous workflow for inductive collaborative qualitative analysis with large language models. In *Proc. CHI Conf. Hum. Fact. Comp. Sys.* (New York, NY, USA, 2024), Mueller F. F., Kyburz P., Williamson J. R., Sas C., Wilson M. L., Dugas P. T., Shklovski I., (Eds.), ACM, pp. 1–29. doi:10.1145/3613904.3642002. 1
- [GLYH10] GUPTA M., LI R., YIN Z., HAN J.: Survey on social tagging techniques. *ACM SIGKDD Explorations Newsletter* 12, 1 (2010), 58–72. doi:10.1145/1882471.1882480. 1
- [GOM18] GANJI A., ORAND M., McDONALD D. W.: Ease on down the code: Complex collaborative qualitative coding simplified with ‘code wizard’. *Proc. ACM Hum.-Comp. Int.* 2, CSCW (2018), 1–24. doi:10.1145/3274401. 2
- [GSS68] GLASER B. G., STRAUSS A. L., STRUTZEL E.: The discovery of grounded theory; strategies for qualitative research. *Nursing Research* 17, 4 (1968), 364. 1
- [HMF*22] HONG M.-H., MARSH L. A., FEUSTON J. L., RUPPERT J., BRUBAKER J. R., SZAFIR D. A.: Scholastic: Graphical human-ai collaboration for inductive and interpretive text analysis. In *Proc. 35th Ann. ACM Symp. User Interf. Soft. Techn.* (New York, NY, USA, 2022), Agrawala M., Wobbrock J. O., Adar E., Setlur V., (Eds.), ACM, pp. 1–12. doi:10.1145/3526113.3545681. 1
- [JWFB21] JIANG J. A., WADE K., FIESLER C., BRUBAKER J. R.: Supporting serendipity: Opportunities and challenges for human-ai collaboration in qualitative analysis. *Proc. ACM Hum.-Comp. Int.* 5, CSCW1 (2021), 1–23. doi:10.1145/3449168. 1, 4
- [Kei00] KEIM D. A.: Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Trans. Vis. Comp. Graph.* 6, 1 (2000), 59–78. doi:10.1109/2945.841121. 2
- [KR24] KANG D., RZESZOTARSKI J. M.: Challenges and opportunities for tool adoption in industrial ux research collaborations. *Proc. ACM Hum.-Comp. Int.* 8, CSCW2 (2024), 1–27. doi:10.1145/3686982. 1, 2, 4
- [Kri19] KRIPPENDORFF K.: *Content analysis: An introduction to its methodology*, 4th edition ed. SAGE Publications, Inc, Los Angeles, CA, 2019. 2, 3
- [MT18] MARATHE M., TOYAMA K.: Semi-automated coding for qualitative research. In *Proc. 2018 CHI Conf. Hum. Fact. Comp. Sys.* (New York, NY, USA, 2018), Mandryk R., Hancock M., Perry M., Cox A., (Eds.), ACM, pp. 1–12. doi:10.1145/3173574.3173922. 4
- [MWA*21] MULLER M., WOLF C. T., ANDRES J., DESMOND M., JOSHI N. N., ASHTORAB Z., SHARMA A., BRIMIJOIN K., PAN Q., DUESTERWALD E., DUGAN C.: Designing ground truth and the social life of labels. In *Proc. 2021 CHI Conf. Hum. Fact. Comp. Sys.* (New York, NY, USA, 2021), Kitamura Y., Quigley A., Isbister K., Igarashi T., Björn P., Drucker S., (Eds.), ACM, pp. 1–16. doi:10.1145/3411764.3445402. 1
- [OSDR24] OVERNEY C., SALDÍAS B., DIMITRAKOPOULOU D., ROY D.: SenseMate: An Accessible and Beginner-Friendly Human-AI Platform for Qualitative Data Analysis. In *Proc. 29th Int. Conf. Intel. User Interf.* (2024), ACM Digital Library, Association for Computing Machinery, pp. 922–939. doi:10.1145/3640543.3645194. 1
- [RM21] RIETZ T., MAEDCHE A.: Cody: An ai-based system to semi-automate coding for qualitative research. In *Proc. 2021 CHI Conf. Hum. Fact. Comp. Sys.* (New York, NY, USA, 2021), Kitamura Y., Quigley A., Isbister K., Igarashi T., Björn P., Drucker S., (Eds.), ACM, pp. 1–14. doi:10.1145/3411764.3445591. 1
- [RPA*24] RICHER G., PISTER A., ABDELAAL M., FEKETE J.-D., SEDLMAIR M., WEISKOPF D.: Scalability in visualization. *IEEE Trans. Vis. Comp. Graph.* 30, 7 (2024), 3314–3330. doi:10.1109/TVCG.2022.3231230. 2
- [Shn83] SHNEIDERMAN: Direct manipulation: A step beyond programming languages. *Computer* 16, 8 (1983), 57–69. doi:10.1109/mc.1983.1654471. 4
- [XYX*19] XIANG S., YE X., XIA J., WU J., CHEN Y., LIU S.: Interactive correction of mislabeled training data. In *2019 IEEE Conf. Vis. Analyt. Sci. Tech.* (2019), Chang R., Keim D., Maciejewski R., (Eds.), IEEE, pp. 57–68. doi:10.1109/VAST47406.2019.8986943. 1
- [ZNX*23] ZHANG Z., NING Z., XU C., TIAN Y., LI T. J.-J.: Peanut: A human-ai collaborative tool for annotating audio-visual data. In *Proc. 36th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2023), Follmer S., Han J., Steimle J., Riche N. H., (Eds.), ACM, pp. 1–18. doi:10.1145/3586183.3606776. 1