Ways of Qualitative Coding: A Case Study of Four Strategies for Resolving Disagreements

Bonnie Chinh
University of Washington, USA
bonniechinh@gmail.com

Himanshu Zade
University of Washington, USA
himanz@uw.edu

Abbas Ganji
University of Washington, USA
ganjia@uw.edu

Cecilia Aragon
University of Washington, USA
aragon@uw.edu

ABSTRACT

The process of qualitative coding often involves multiple coders coding the same data to ensure reliable codes and a consistent understanding of the codebook. One aspect of qualitative coding includes resolving disagreements, where coders discuss differences in coding to reach a consensus. We conduct a case study to evaluate four strategies of disagreement resolution and understand their impact on the coding process. We find that an *open discussion* and the *n-ary tree metric* lead coders to focus more on the disagreement of a particular data instance, whereas *kappa values* and *Code Wizard* direct coders to compare code definitions. We discuss opportunities for using different strategies at different stages of the coding process for more effective disagreement resolution.

KEYWORDS

Disagreement; qualitative coding

CHI'19 Extended Abstracts, May 4–9, 2019, Glasgow, Scotland Uk

ACM ISBN 123-4567-24-567/08/06.
https://doi.org/10.475/123_4

Strategies of Disagreement Resolution
Open discussion: no defined method on how to approach disagreements (baseline)
Kappa values: focus on codes where kappa values are lowest
N-ary tree metric: codes sorted from lowest to highest agreement (Zade et al. [15])
Code Wizard: focus on codes based on correlated certainty and correlated disagreement (Ganji et al. [6])

INTRODUCTION

Qualitative research often involves interpreting rich data and extracting the core concepts of that data as concise points. The process to understand the data is often done through qualitative coding, where multiple coders individually assign codes from a codebook to data instances. Disagreements on which codes apply to a data instance often occur during this process—these are necessary to resolve in order to gain a collective understanding of the codebook and apply the codebook consistently.

In this work, we conduct a case study of four strategies used to guide disagreement resolution in coding. We use an open discussion as a baseline, kappa values as a common discussion tactic, and the n-ary tree metric and Code Wizard as recent state-of-the-art strategies [6, 15]. For each evaluation, coders independently code a set of data and use a different strategy to resolve disagreements.

We document the experience of using each strategy, identify their strengths, and reflect on their impact in disagreement resolution. The lessons we share are important for understanding the ways of resolving disagreements in qualitative coding. Moreover, it is important for the design of future software to support the process of qualitative coding and disagreement resolution.

RELATED WORK

Collaborative qualitative coding is intended for two main purposes: (1) to provide a sound interpretation of data in response to the challenge of the subjectivity of qualitative data [1], and (2) to infer reliability from the observed agreement among independent coders [10]. This iterative process often ends by reaching an acceptable agreement threshold. Traditionally, inter-coder agreement is achieved by consensus in discussion meetings or by evaluating the inter-coder reliability (ICR) coefficient [3, 11].

Investigating inter-coder disagreement has attracted much attention in the last decade. Krippendorff distinguishes systematic and random disagreement in collaborative coding [9]. Moreover, researchers argue that systematic disagreement threatens reliability of analysis even if the ICR coefficient is high [9, 12]. Another perspective on inter-coder disagreement focuses on disagreement as a tool to improve and accelerate the process towards an agreement threshold. It views systematic disagreement as predictable, correctable, and containing hidden but helpful information [2, 6, 8, 9, 14].

Many commercial platforms have been developed for qualitative analysis such as NVivo, Atlas.ti, Dedoose, and HyperResearch. While these platforms enable users to analyze multiple data types (e.g., text, image, voice), they do not sufficiently facilitate coders to recognize sources of disagreement [6]. Recently, several computer-assisted tools have been developed to find sources of disagreement. CrowdTruth is open-source software for collaborative data annotation [7]. Aeonium flags ambiguous instances using machine learning models [5]. Some research also uses crowdsourcing to train machine learning datasets [4] or predict levels of disagreements using a deliberation workflow for crowdworkers [13]. Zade et al. conceptually frame disagreements in terms of diversity and divergence

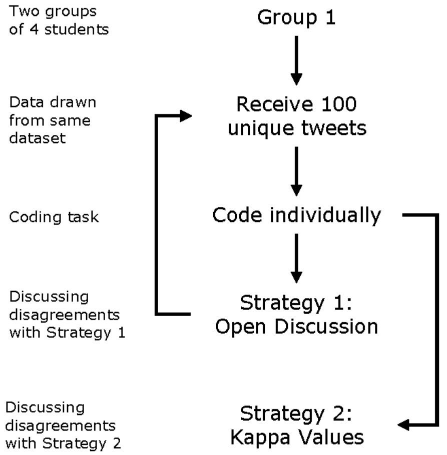

Two groups of 4 students
Group 1
Data drawn from same dataset
Receive 100 unique tweets
Coding task
Code individually
Discussing disagreements with Strategy 1
Strategy 1: Open Discussion
Discussing disagreements with Strategy 2
Strategy 2: Kappa Values

Figure 1: The diagram demonstrates the workflow of coding and discussing disagreements for Group 1. A similar workflow was followed for Group 2, but with different strategies.

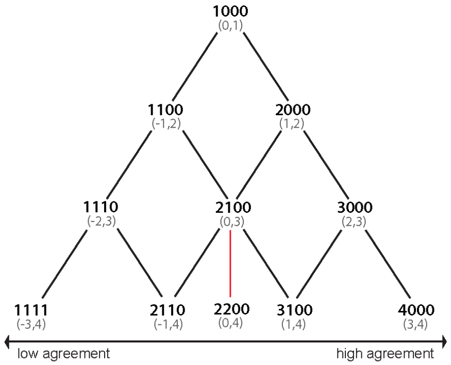


Figure 2: The n-ary tree metric that determines low agreement to high agreement for four coders.

and propose two tree-based metrics to compare them. [15]. Code Wizard produces correlated coders' uncertainty and correlated disagreement matrices to point out ambiguous and conflicting codes [6].

METHODS

We pursued this research as a part of a larger project in which eight students used a grounded theory approach to code 4200 tweets related to data privacy across 10 weeks. Our case study began in the fifth week after collectively creating an early codebook draft and followed the workflow illustrated in Figure 1. Each group had one expert in qualitative coding and three with little to no prior experience.

We had a total of four one-hour meetings to discuss coding disagreements, each using a different strategy for disagreement resolution and led by an experienced member of the group. Each discussion leader followed a common protocol to introduce the strategy, moderate the discussion, and document the experience from a leader's perspective. We administered a survey after each meeting to gather anonymous feedback on the strategy used for resolving disagreements.

Coding Discussion Strategies

Strategy 1: Open Discussion (Group 1). The discussion leader created a summative table to display which code was used for each tweet by each of the four coders. The only instructions were to discuss and resolve all instances that did not have complete consensus. This strategy acted as a baseline to observe how the coders would choose to resolve disagreements.

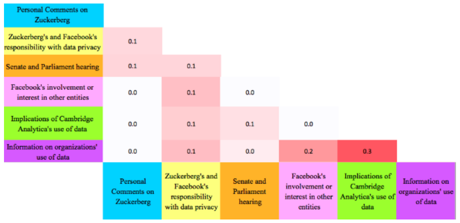
The coders chose to resolve every instance of disagreement in the order they appeared in the summative table. Coders took notice of which codes were used by others for a particular tweet and listened to each person's rationale for using a code before making a decision.

Strategy 2: Kappa Values (Group 1). Prior to this meeting, the discussion leader calculated the overall Fleiss' kappa for the four coders. Additionally, kappas for each code were calculated. This information, along with a summative table displaying the codes used per tweet, served as a guide for the discussion.

In the meeting, the discussion leader helped coders interpret the meaning of the kappa values. Coders discussed disagreements beginning with the code with the lowest kappa values. The coders discussed every tweet involving that code until consensus before focusing on another code.

Strategy 3: N-ary Tree Metric (Group 2). Tweets were organized using the n-ary tree metric from lowest to highest agreement [15]. The tree-based metric shows the number of coders in its depth and distributions of agreement on each level sorted from low to high. The discussion leader created a summative table in which the coders were also able to filter to a specific level of the n-ary tree.

Coders began reviewing all instances of low agreement {1111} before moving to levels of higher agreement {4000} (see Figure 2). At each instance, coders discussed the rationale behind the code they assigned until they reached a consensus about which code best suits the tweet instance.



Personal Comments on Zuckerberg						
Zuckerberg's and Facebook's responsibility with data privacy	0.1					
Senate and Parliament hearing	0.1	0.1				
Facebook's involvement or interest in other entities	0.0	0.1	0.0			
Implications of Cambridge Analytica's use of data	0.0	0.1	0.1	0.0		
Information on organizations' use of data	0.0	0.1	0.0	0.2	0.3	
	Personal Comments on Zuckerberg	Zuckerberg's and Facebook's responsibility with data privacy	Senate and Parliament hearing	Facebook's involvement or interest in other entities	Implications of Cambridge Analytica's use of data	Information on organizations' use of data

Figure 3: Correlated disagreement matrix generated by Code Wizard.

Strategy 4: Code Wizard (Group 2). Coders inputted their codes into Code Wizard prior to the meeting to generate tables for resolving disagreements [6]. Code Wizard offers perspectives of understanding codes through coder certainty, correlated uncertainty, and correlated disagreement.

Coders referred primarily to the correlated disagreement matrix (see Figure 3) to identify code-pairs which were used to code the same data instance, and resolved all instances which contained those code-pairs. Thus, the disagreement-coefficients guided the disagreement resolution process.

EVALUATION OF STRATEGIES

Each group of coders coded two sets of data (four total) and participated in two disagreement resolution meetings using different strategies. We evaluate these strategies based on survey questions about their perceived usefulness or productivity. Additionally, we discuss the documentation and observations from the discussion leaders on the perceived success of the strategy used in each meeting.

Comparing Strategies in Group 1: Open Discussion and Kappa Values

When coders used an open discussion, they chose to resolve all instances in the order that they appeared in a summative sheet. All coders believed that this was an effective way to understand the codes and disagreements. They also believed that the pace of resolving disagreements was productive.

Kappa values were used to identify codes that resulted in more disagreement. Two out of four coders believed it was effective, one coder felt neutral, and another coder felt that it was not effective. Their opinions followed the same distribution for productivity.

Coders agreed that using kappa values was a useful strategy for discussing codes in general, but did not believe it was more effective than having an open discussion. They noted that their discussions were more focused on code ambiguity which was helpful for understanding codes. *"[Kappas] seemed to make more sense to me because it allowed us to spend our time focusing on the codes we had the most disagreement about to get at the underlining misunderstandings between coders rather than going through randomly."* The pace of resolving disagreements was similar in both strategies. Coders resolved 17 disagreements with an open discussion and 16 disagreements with kappa values (see Table 1).

Table 1: Number of disagreements that were resolved for each strategy. "Total" is the total number of disagreements that occurred in a set of 100 tweets.

	Coding Strategy	Resolved
Group 1	Open Discussion	17 of 54
	Kappa Values	16 of 49
Group 2	N-ary Tree Metric	21 of 61
	Code Wizard	10 of 50

Comparing Strategies in Group 2: The N-ary Tree Metric and Code Wizard

The n-ary tree metric was used to sort levels of agreement. Three out of four coders felt that the strategy was effective, whereas one coder felt neutral about its effectiveness. Most believed the n-ary tree metric was *"helpful to identify instances that we know need more time to discuss given the spread of labels across different codes by all the coders."*

All coders believed that Code Wizard was an effective tool for resolving disagreements. Three out of four coders thought that using Code Wizard was more effective than using the n-ary tree metric, while one coder found it less effective due to its complexity.

Coders found that Code Wizard took more time to understand because of its multiple visualizations and tables, but overall gained more insight into why coders may choose different codes. *"Although this strategy took more time and was more complex, I found that the detail is super helpful to highlight which tweets were more divisive than others . . . it had more meaning, or tackled the disagreement more in depth than the previous strategy."* However, they believed that the complexity caused them to be less productive in pace, resolving fewer disagreements with Code Wizard than the n-ary tree metric. Table 1 shows that the coders resolved 21 disagreements when using the n-ary tree metric compared to 10 disagreements when using Code Wizard. This demonstrates a strong difference in the pace of resolving disagreements using the n-ary tree metric compared to Code Wizard.

Reflections of Discussion Leaders

The open discussion strategy allows coders to quickly assess coding disagreements in chronological order, which led to more focus on each data instance, rather than the codebook, as observed by the discussion leader. The n-ary tree metric enables coders to sort through disagreements and affords coders the ability to find instances of strongest disagreement. The discussion leader noted that when they faced stronger disagreement levels, discussions took longer and it was harder to come to a consensus. Both leaders observed that kappa values and Code Wizard led coders to have discussions with more depth, focusing on the most ambiguous codes and reconsidering where each code applies. One observed that the kappa values were *"useful and necessary for getting everyone on the same page"* to have the same understanding of the codebook. Another discussion leader observed that Code Wizard also proved useful in disambiguating codes, but involved a higher learning curve.

DISCUSSION

Each coding strategy can have unique strengths that are used towards resolving disagreements. Its usefulness may also be dependent on the state of the codebook as considered by the coders. Through our case study, it is evident that strategies which focus on code ambiguity (e.g. correlated disagreement matrix in Code Wizard or kappas by code) can help early stages of qualitative coding by disambiguating codes, but may require more time to resolve disagreements. On the other hand, strategies that focus primarily on disagreements of data instances (e.g., n-ary tree metric) may be more useful in later stages of coding, where a robust codebook has already been established.

The findings of our research are limited given that the perspectives come from eight coders and can be prone to personal bias or order effects. Each group of coders only tested two strategies due to resource limitations so we could not compare across all strategies. Still, our research suggests it may be beneficial to combine multiple strategies to resolve disagreements rather than focus on one. Software supporting this process should consider both the present state of disagreements and the confidence coders have in their codebook to devise an appropriate strategy for resolving disagreements.

Qualitative researchers often use a collaborative coding process to identify prominent themes in the data. Meanwhile, some researchers use the coded data towards techniques for large scale analysis. Given this tension, it would be useful to employ multiple strategies at different stages of coding to both improve codebook understanding and increase the pace of resolving disagreements.

REFERENCES

[1] Rosaline S Barbour. 2001. Checklists for improving rigour in qualitative research: a case of the tail wagging the dog? *BMJ: British Medical Journal* 322, 7294 (2001), 1115.

[2] Eyal Beigman and Beata Beigman Klebanov. 2009. Learning with annotation noise. *In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP* 1 (2009), 280–287.

[3] John L Campbell, Charles Quincy, Jordan Osserman, and Pedersen Ove K. 2013. Coding In-depth Semistructured Interviews: Problems of Unitization and Intercoder Reliability and Agreement. *Sociological Methods and Research* 42, 3 (2013), 294–320.

[4] Joseph Chee Chang, Saleema Amershi, and Ece Kamar. 2017. Revolt: Collaborative crowdsourcing for labeling machine learning datasets. *In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (2017), 2334–2346.

[5] Margaret Drouhard, Nan-Chen Chen, Jina Suh, Rafal Kocielnik, Vanessa Pena-Araya, Keting Cen, Xiangyi Zheng, and Cecilia R. Aragon. 2017. Aeonium: Visual analytics to support collaborative qualitative coding. *In Pacific Visualization Symposium (PacificVis), IEEE 2017* (2017), 220–229.

[6] Abbas Ganji, Mania Orand, and David W McDonald. 2018. Ease on Down the Code: Complex Collaborative Qualitative Coding Simplified with'Code Wizard'. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 132.

[7] Oana Inel, Khalid Khamkham, Tatiana Cristea, Anca Dumitrache, Arne Rutjes, Jelle van der Ploeg, Lukasz Romaszko, Lora Aroyo, , and Robert-Jan Sips. 2014. Crowdtruth: Machine-human computation framework for harnessing disagreement in gathering annotated data. *In International Semantic Web Conference* (2014), 486–504.

[8] Emily Jamison and Iryna Gurevych. 2015. Noise or additional information? Leveraging crowdsource annotation item agreement for natural language tasks. *In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (2015), 291–297.

[9] Klaus Krippendorff. 2008. Systematic and random disagreement and the reliability of nominal data. *Communication Methods and Measures* 2, 4 (2008), 323–338.

[10] Klaus Krippendorff. 2011. Agreement and Information in the Reliability of Coding. *Communication Methods and Measures* 5, 2 (2011), 93–112.

[11] Catherine MacPhail, Nomhle Khoza, Laurie Abler, and Meghna Ranganathan. 2016. Process guidelines for establishing Intercoder Reliability in qualitative studies. *Qualitative research* 16, 2 (2016), 198–212.

[12] Dennis Reidsma and Jean Carletta. 2008. Reliability measurement without limits. *Computational Linguistics* 34, 3 (2008), 319–326.

[13] Mike Schaekermann, Joslin Goh, Kate Larson, and Edith Law. 2018. Resolvable vs. Irresolvable Disagreement: A Study on Worker Deliberation in Crowd Work. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW 2018 (2018), 154.

[14] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. 2015. Learning from massive noisy labeled data for image classification. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), 2691–2699.

[15] Himanshu Zade, Margaret Drouhard, Bonnie Chinh, Lu Gan, and Cecilia Aragon. 2018. Conceptualizing Disagreement in Qualitative Coding. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 159.