Conceptualizing Disagreement in Qualitative Coding

Himanshu Zade himanz@uw.edu
Margaret Drouhard meg.drouhard@gmail.com
Bonnie Chinh bchinh@uw.edu
Lu Gan ganlu@uw.edu
Cecilia Aragon aragon@uw.edu

Human Centered Design and Engineering, University of Washington, Seattle, WA.

ABSTRACT

Collaborative qualitative coding often involves coders assigning different labels to the same instance, leading to ambiguity. We refer to such an instance of ambiguity as disagreement in coding. Analyzing reasons for such a disagreement is essential– both for purposes of bolstering user understanding gained from coding and reinterpreting the data collaboratively, and for negotiating user-assigned labels for building effective machine learning models. We propose a conceptual definition of collective disagreement using *diversity* and *divergence* within the coding distributions. This perspective of disagreement translates to diverse coding contexts and groups of coders irrespective of discipline. We introduce two tree-based ranking metrics as standardized ways of comparing disagreements in how data instances have been coded. We empirically validate that, of the two tree-based metrics, coders' perceptions of disagreement match more closely with the *n-ary* tree metric than with the *post-traversal* tree metric.

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

Author Keywords

Disagreement; ambiguity; qualitative coding; theory.

INTRODUCTION

Qualitative researchers work to capture rich insights from human data, but as we generate ever-increasing quantities of data, focusing researchers' efforts on the most *interesting* or *significant* information comprises a major challenge. Often the most interesting information may be subtle, ambiguous, or raise disagreement among coders. Consider a case where researchers want to analyze perceptions about political views based on a data set of tweets. They want to apply five mutually exclusive codes to each tweet: support, rejection, neutral, unrelated, and uncodable. Given the enormous data set and time taken for manual coding, it is likely that the researchers may have some partially coded tweets. They will also have disagreements on many of these tweets, too many to spend time discussing face-to-face in a group. How can we sort all the ambigious tweets from most ambiguous to least ambigious? Such a sorting technique will allow qualitative researchers who may not be computer scientists to focus on data instances that are most challenging or confusing to code.

Exploring potential disagreements in more depth is often necessary in qualitative coding, but it poses many challenges. First, it is difficult to reach absolute agreement since qualitative coding relies on subjective judgments. Second, mapping or estimating degree of disagreement along a numeric scale requires shared understandings about the nature or dimensions of disagreement in a specific context. The variety of possible contexts and project values present a challenge to developing a metric that is applicable across disparate qualitative coding schemes. In order to appropriately rank collective disagreement in collaborative qualitative coding, we consider the *diversity* of conflicting codes and the *divergence* or "strength" of a disagreement as indicated by number of coders who applied differing codes. In this work, we:

- Offer a conceptual definition of collective disagreement that translates to diverse coding contexts and groups of coders irrespective of discipline.
- Contribute two standardized metrics for ranking disagreement.
- Evaluate how well these metrics align with the intuitions of qualitative coders who consider disagreement as one factor in their effort to be more conscious about how codes are applied.

In the following sections, we (1) provide background on the contexts for disagreement through two case studies, (2) introduce related research about the articulation and representation of disagreement, (3) explain our conceptual framework of tree-based ranking metrics, and (4) present findings from our empirical validation of the metrics and a visual representation of them. Finally, we discuss the significance of our findings with respect to disagreement in qualitative coding and other disciplines, and we outline directions for further exploration.

BACKGROUND

Qualitative researchers may have a variety of diverse objectives in qualitatively coding data. While some researchers aim to improve coding consistency for building accurate models, other researchers focus not on seeking consensus, but rather on building deep understanding through consideration of different perspectives. Programmatic analysis of qualitative coding is further complicated for researchers by the exploratory goal of qualitative research, fundamental differences between quantitative and qualitative research methods, low accuracy of qualitative coding, and unfamiliarity of qualitative researchers with machine learning (ML) techniques [10]. Although ML has thrived in the past decades, there are only limited applications

CHI 2018, April 21–26, 2018, Montréal, QC, Canada.

http://dx.doi.org/10.1145/3173574.3173733

in qualitative analysis to facilitate the coding process for labeling large datasets using fully or semi-automatic methods [30]. Moreover, the inherently subjective nature of qualitative coding results in inconsistencies among different coders, further impacting the quality of coding.

It is common for different qualitative coders to disagree about the appropriateness of a code used to label a data instance. When several collaborative qualitative coders differ from each other in their codes, their disagreement may ambiguate the group's overall understanding of that data instance. Disagreement may also increase complexity for building a ground truth dataset and training strong ML models for qualitative coding. Therefore, it is essential that we address disagreement and focus on identifying points of probable inconsistency in the context of qualitative analysis. Foundations of disagreement may include unclear code definitions or particular data points that collaborators need to negotiate or clarify. Alternatively, disagreement amongst qualitative coders may draw from their diverse experiences, priorities, and ways of communicating [4]. While it may be challenging to explore and build understanding around disagreements, the process often yields unanticipated insights from the data. Improving methods for dealing with disagreements can therefore improve consistency and collective understanding in coding while retaining the rich perspectives of diverse coders.

To illustrate some of the ways in which qualitative coding might be improved through better processes for recognizing and addressing disagreement, we present insights from two case studies in the following sections.

Case Study I

In order to better understand the role disagreement plays in qualitative coding, we organized an open-ended discussion with a qualitative researcher in order to understand all aspects of the coding process and how they could be improved. While describing one of her ongoing projects, the researcher mentioned the use of disambiguation techniques in her research, which primarily consisted of discussion among coders to derive a consensus and the use of an impartial arbitrator when coders could not discuss to consensus. The researcher, who often acted as the arbitrator in her work, also expressed concern whether her own seniority subjected coders to agree with her views or biased coders' decisions.

The use of an arbitrator to resolve conflicts amongst qualitative coders is a standard norm that utilizes the authority of an individual coder over others involved in the process. We argue that the coding process would often benefit from distributed authority for arbitration, rather than privileging one researcher's perspectives over others. Rather than rely on a single individual's judgment, we propose that a better authority might be a method or process that includes thoughtful consideration of disagreements and values a multiplicity of perspectives.

Case Study II

Disagreements grow more complex as the number of coders increases. In consideration of this elevated complexity, we present another research study in which 3-8 coders coded each data instance [22]. In this work, researchers contributed visual analytics tools to provide overviews of codes applied to very large datasets. For a dataset of 485,045 text instances, coders were able to code around 5% in eight weeks. Visualization is well-suited to human pattern-finding tasks in large datasets, but may still be challenging to interpret in cases of very large datasets or large numbers of coders. The introduction of metrics that automatically sort ambiguous instances of coded data from most ambiguous to least ambiguous will enable researchers to build better visualizations. By empowering coders with the ability to sort by coder disagreement, such metrics will also be useful in reducing human efforts of disambiguation as in this case study.

Diversity & Divergence in Conceptualizing Disagreement

Disagreement can be characterized through many dimensions. In the scope of our research, we discuss two dimensions—*diversity* and *divergence*. Diversity refers to the variation of codes, or *labels*, used by different coders, where a larger distribution indicates greater disagreement (e.g., if four coders use four different labels, there is high diversity). Divergence, on the other hand, refers to clusters of coders agreeing on different labels and thus diverging from each other (e.g. if two coders choose label 1 and another two coders choose label 2, there is high divergence). Diversity and divergence are not mutually exclusive; high diversity is not devoid of divergence, although low divergence may be seen as a less important or secondary factor when compared to a situation of high diversity. In the remainder of this work, we explore shades of disagreement through the lenses of diversity and divergence in order to determine which metric(s) for disagreement might be most useful for qualitative research.

RELATED WORK

Qualitative coding may be viewed as a set of methodologies to impose structure on unstructured data through the application of "codes," or analytical labels to related instances of data [28, 24]. Given the lack of intrinsic structure in the data, the quality of coding may significantly impact researchers' analysis of the coding scheme [29]. In this section, we provide an overview of prior work evaluating qualitative coding methods, and offer discuss in detail the role that understanding disagreement has played in qualitative coding.

Inter-Rater Reliability and Disagreement

The "quality" of coding is subjective and may signify different types and levels of validation in various contexts, but one commonly applied measure to assess coding consistency between multiple coders is inter-rater reliability (IRR), frequently calculated using *Cohen's Kappa* [6]. Calculating IRR is a form of quantitative analysis, but has been shown to impact coding quality in a number of cases [16, 20, 3].

IRR is the measurement of the extent to which coders assign similar scores to the same variable) [25]. The IRR, i.e., degree of agreement amongst different coders may be used to ensure that the reported data is an actual representation of the variables that are measured. To overcome the occasionally unexpected results yielded by pi (π) and kappa (κ) statistics, which are most widely used for testing the degree of agreement

between raters, researchers have devised alternate agreement statistics to account for the role randomness plays in agreement between coders [18]. Even given meticulous coding on the part of all coders, discerning the significance of various states and degrees of disagreement may be challenging. IRR metrics such as Cohen's Kappa, Cronbach's Alpha, and others evaluate inter-rater agreement over a set of data, which may be most useful for evaluating consistency of coding at a grand scale. In contrast, our metrics rank the degree of disagreement on particular data instances, which allows for consideration of disagreement at an instance-level. We provide more discussion about the significance of forms and degrees of disagreement in later sections.

Consensus

One of the most common techniques for addressing disagreement in qualitative coding is to negotiate toward consensus. In other words, individuals make estimations and negotiate their response before reporting the final answer [12]. Prior attempts to realize consensus include use of interaction and visualization features like distributed design discussions for bringing consensus strategies to unmoderated settings [13]. Armstrong et al. have demonstrated through empirical qualitative techniques that while researchers indicate close agreement upon basic themes, they may report different understandings from those similar themes [1]. These findings align with the results of our expert evaluation where participants indicated different processes (either prioritizing diversity or divergence) to infer the same ranking of disagreement.

Diversity and Divergence in Coding and Annotation

When it comes to examining variance in qualitative coding, most observational research only assesses agreement, while reliability is assumed given sufficient agreement. Measuring agreement can: (i) indicate the trustworthiness of observations in the data, and (ii) provide feedback to calibrate observers against each other or against baseline observations. If one assumes the baseline to be 'true', then observer agreement can be used to assess reliability. As we described above, a commonplace statistic to assess observer agreement, Cohen's Kappa [6], evaluates consistency of coding at a grand scale. For instance-level analysis, we now discuss techniques that learn from coder variation and harness the diversity of opinions for improving coding results.

Systematic divergence

Kairam and Heer introduce "crowd parting," a technique that clusters sub-groups of crowd workers whose annotations systematically diverge from those of other sub-groups [21]. They applied this technique to crowd-worker annotations, and identified several themes that may lead to systematically different, yet equally valid, coding strategies: *conservative vs. liberal annotators*, *label concept overlap*, and *entities as modifiers*. Sub-groups identified by crowd parting have internally consistent coding behavior though their coding decisions diverge from the plurality of annotators at least for some subsets of the data.

Disagreement is signal not noise

Disagreement has long been viewed as an hinderance to the practice of qualitative coding. However, researchers recently have challenged this view. Through their experimental work on human annotation, Aroyo et al. have debunked multiple "myths" including the ideas that one valid interpretation, or a single ground truth, should exist and that disagreement between coders indicates a problem in coding [2]. Their research points out that disagreement can signal ambiguity; systematic disagreements between individuals may also indicate multiple reasonable perspectives. Lasecki et al.'s work demonstrates the value of both measuring the signal of disagreement, and ranking it among coders [23]. In their tool *Glance*, they measure coder disagreement using inter-rater reliability and variance between coders labels to facilitate rapid and interactive exploration of labeled data, and to help identify problematic or ambiguous analysis queries. Our metrics provide an alternate method to sort and filter disagreements, taking into account the potential significance of a variety of different states of disagreement.

Probing disagreement and ambiguity

Several efforts have attempted to utilize the diversity of opinions for further improving the coding results. The MicroTalk system successfully exploits crowd diversity by presenting counterarguments to crowdworkers to improve the overall quality of coding [9]. Another example of such a system is *Revolt*, a platform that allows crowdworkers to provide requesters with a conceptual explanation for ambiguity when disagreements between coders occurred. Chang et al. demonstrate that meaningful disagreements are likely to exist regardless of the clarity or specificity provided in coding guidelines, and that probing these disagreements can yield useful insights as to ambiguity within data instances, coding guidelines or both [5]. While investigating the scope of using disagreement between annotators as a signal to collect semantic annotation, researchers have confirmed the need and potential to define metrics that capture disagreement [11]. This illustrates the value of characterizing and probing disagreements among qualitative coders. Our approach is built upon the same premise, but our metrics are based on distributions of codes and place value on different dimensions of disagreement: diversity and divergence.

TREE-BASED RANKING METRICS FOR DISAGREEMENT

As more coders assign labels to a data instance, their responses can either match or conflict with labels assigned by other coders. We refer to the distribution of labels at any stage of coding as a *state of agreement*. Each new coder labeling the data instance generates a new state of agreement. For example, if all coders assign the same labels to a data instance, it generates a state of complete agreement, i.e., no disagreement. Each additional coder either strengthens the state of complete agreement (by supporting the same label) or weakens it (by choosing a different label). Therefore, a data instance will enter a new state of agreement with each additional label, regardless of whether the label matched or conflicted with existing labels.

Given m mutually exclusive labels and n coders, we define a metric for ranking agreement between coders. A tuple of length m represents the distribution of codes assigned by n coders across the m labels. For example, a tuple $\{310\}$ represents the distribution of 4 codes from 4 coders across 3 labels. In this scheme,

1. Each element of the tuple represents the number of coders agreeing upon a unique label. For example, a tuple $\{310\}$ represents 3 coders agreeing on one label, 1 coder assigning second label, while no coder assigned the third label.
2. The tuple is ordered such that labels upon which more coders agree are sorted left and labels for which there is less agreement are sorted to the right. This means that the label chosen by the highest number of coders will always be the first element of the tuple, and the label chosen by the least number of coders will always be the last element of the tuple, *e.g.*, $\{310\}$.
3. Since our approach depends entirely on the distributions of agreed-upon codes, the order of the tuple is not relevant for ranking, but provides clarity in explaining the metric.

Next, we present a brief overview of the fundamental principles behind our two tree-based metrics proposed for ranking disagreement. We describe them in detail in a later section.

1. *The post-traversal tree ranking (Figure 1):* This tree metric focuses on identifying a coder-group of the maximum possible size which collectively opposes another label chosen by a majority of coders. We utilize dynamic programming techniques to structure the nodes such that all states of agreement involving a group of maximum possible size forms the left sub-tree, while remaining states of agreement are organized under the right sub-tree. A simple postorder traversal (or *post-traversal*) ranks the disagreement nodes from less to more agreement.
2. *The n-ary tree ranking (Figure 2):* This tree metric extends upon a previous state of agreement by considering all the possible labels which a coder can choose. Thus, each connected child-node represents different possibilities for coding that data instance as chosen by the next coder. While some of the states clearly suggest more agreement amongst the coders, a few states suggest ambiguity between two states of disagreement. Our ranking algorithm identifies such instances of ambiguous degrees of disagreement and ranks them based on number of coders who have labeled the instance.

Post-Traversal Tree Metric of Disagreement

A top-down view of disagreement can be formulated by considering how a majority agreement about coding a data instance can be challenged by another group of coders. In other words, given an agreement between n coders about labeling a data instance, we sequentially explore the possibility of different $n, n-1, n-2 \ldots 2, 1$ coders offering an alternate label for the same instance. When expressed thus, the problem of computing the different combinations is reducible to the coin change dynamic programming problem [1]. The set of all possible group sizes in which coders can collectively agree amongst themselves correspond to the set of available coins, while the number of coders yet to assign labels corresponds to the total amount of change required.

[1] https://en.wikipedia.org/wiki/Change-making_problem

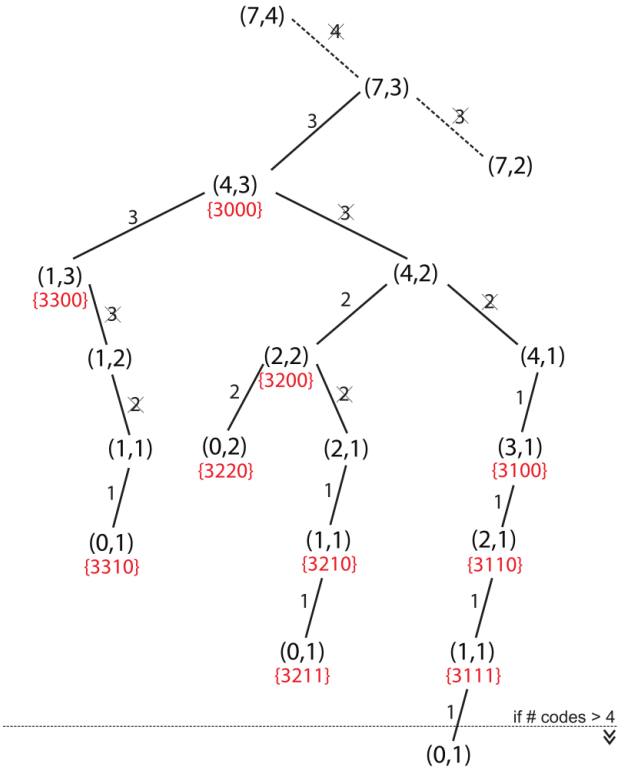

Figure 1. The ranking metric as conceptualized by the post-traversal tree. The 4-length tuples represent the different states of agreement, while the encircled numbers alongside indicate the rank of disagreement based on a simple postorder traversal. The lower the rank, the lower the agreement within that state of agreement. NOTE: The metric only consists of a ranking (low to high agreement) between several coding distributions as input by a participant, *e.g.*, order of agreement $\ldots\{3111\} < \{3110\} < \{3100\}\ldots$, and so on. The tree visualization only offers a conceptual understanding into how the metric is operationalized.

We use the notation (C,n), where C is the number of coders yet to assign labels and n is the maximum number of coders who can agree on the same label, to refer to the number of different states of agreement. The notation $(C-n,n)$ then corresponds to the possibilities that n coders agreed to use the same label for coding a data instance at least once. The remaining possibilities include all states of agreement where no $n-1$ coders agreed on a label, i.e., C coders need to assign a label where no more than $n-1$ coders agree. Thus, the possible disagreements represented by (C,n) can be broken down into $(C-n,n)$ and $(C,n-1)$.

Figure 1 gives a glance into some of the different states of coding agreements that can be reached when 7 coders try to assign any of the available labels to a data instance, given the limit that a maximum of 3 coders can agree on any single label. The different states of agreement can then be ranked from low to high agreement using a post-order traversal as

follows: $\{3310\} < \{3300\} < \{3220\} < \{3211\} < \{3210\} < \{3200\} < \{3111\} < \{3110\} < \{3100\} < \{3000\}$.

We use a tuple of length equal to the number of total available labels to represent a state of agreement, where we assign one digit for each of the labels to represent the coding. Let tuple $\{1000\}$ represent a state where one user assigned a label (out of 4 label choices), while other coders have yet to assign any code. Every time the maximum permissible number of coders agree to use the same label, we record the new state of agreement as a coding tuple.

Thus, we record a tuple $\{3300\}$ in Figure 1 to indicate that 2 groups of 3 coders have agreed on 2 distinct labels. Each tuple represents a state of agreement. Ranking these different tuples is reduced to a simple post-order traversal of the tree with the leftmost tuple indicating lowest agreement. Algorithm 1 gives the pseudo-code to recursively compute the rank using this approach for any disagreement. Although this approach does not uncover any more coding combinations than those in the n-ary tree approach introduced in a later section, it supports absolute ranking unlike the n-ary tree based ranking that does not force-rank ambiguous instances.

The dynamic programming based post-traversal tree (Figure 1) is built in a top-down manner. Therefore, this tree prioritizes the maximum size of group that can oppose any existing majority agreement amongst coders over the choice of a label. This chain of thought aligns with high divergence and low diversity. We revisit this thought when we discuss our qualitative study with experts.

Algorithm 1 Postorder traversal based ranking algorithm

```
C ← #Coders
n ← #RequiredLabels
offset ← 0
function RANK(C, n, offset)        ▷ We begin with C = n
    L ← Rank(C − n, n, offset)
    R ← Rank(C, n − 1, L)
    return L + R + 1
```

The N-ary Tree Metric for Disagreement

Consider the case of five qualitative coders labeling a data instance using four labels (A, B, C, & D). Let tuple $\{1000\}$ represent a state of agreement where one user assigned a code A to an instance, while other coders have yet to assign any code. There are only two possible outcomes that a second coder could generate because they will either agree with the previous coder or disagree, assigning a new label. Thus, two possible tuples are available: $\{1100\}$ or $\{2000\}$. The n-ary tree presents the possible coding outcomes as more coders are added. Each tuple is also accompanied by an index i and depth d as (i,d). The index begins at 0 for $\{1000\}$ and increases if agreement is added or decreases if disagreement is added. The depth also begins at 1 and increases by 1 for every new coder, as new coders create new levels of the tree.

With each new coder assigning a label, the n-ary tree (Figure 2) extends a level down and exhausts all the possible states of agreements that could be reached. By considering all the possibilities, n-ary tree ensures it does not bias or favor towards diversity or divergence, but rather preseves the complexity of qualitative coding. We revisit this thought when we discuss our qualitative study with experts.

Algorithm

Our algorithm recursively defines an n-ary tree metric for disagreement, in which the depth of the tree represents the number of coders who labeled a particular data instance. The root of the tree represents one coder, with descending levels adding one coder per level. As such, level d includes all possible distributions of codes that could be chosen by d coders.

The ranking metric is achieved through a branching system that offers up to three choices from each node t at level $i-1$ in the tree:

1. add agreement, representing the ith coder choosing the highest ranked code in node t
2. add disagreement, representing the ith coder choosing a code that had not been chosen in node t; and
3. add both agreement and disagreement, representing the ith coder choosing a code that has been chosen but which is not the highest ranked code in t

Figure 2 illustrates the nodes as recursively assigned coordinates based on these three branch choices. Based on these coordinates, Algorithm 2 describes a pseudo-code to decide the order of agreement. Thus, we demonstrate that it is possible to provide a simple, standardized technique to rank disagreement. In Section 5, we show that our ranking system aligns with participants' intuitive understandings of disagreement.

Algorithm 2 N-ary tree based ranking algorithm

```
function RANK(A, B)                          ▷ A and B are tuples
    a ← getIndex(A)  ▷ Returns tree coordinates of node
    b ← getIndex(B) ▷ In Fig 2, getIndex(1000) = (0, 1)
    if a.col > b.col then                    ▷ If a ← (1, 2), a.col = 1
        Rank_A > Rank_B
    else if a.col < b.col then
        Rank_A < Rank_B
    else
        if abs|a.depth − b.depth| ⩽ 1 then
            Rank_A = Rank_B
        else if a.depth > b.depth then
            Rank_A > Rank_B
        else
            Rank_A < Rank_B
```

MTURK USER STUDY

Study Design

We conducted a user study to validate that the ranking of agreement as proposed by our post-traversal tree and n-ary tree metric aligns with people's perception of disagreement. Rather than measuring *disagreement* in the study, we measure perceptions of *agreement* to avoid negative questions. One of the techniques important for assessing inter-rater reliability

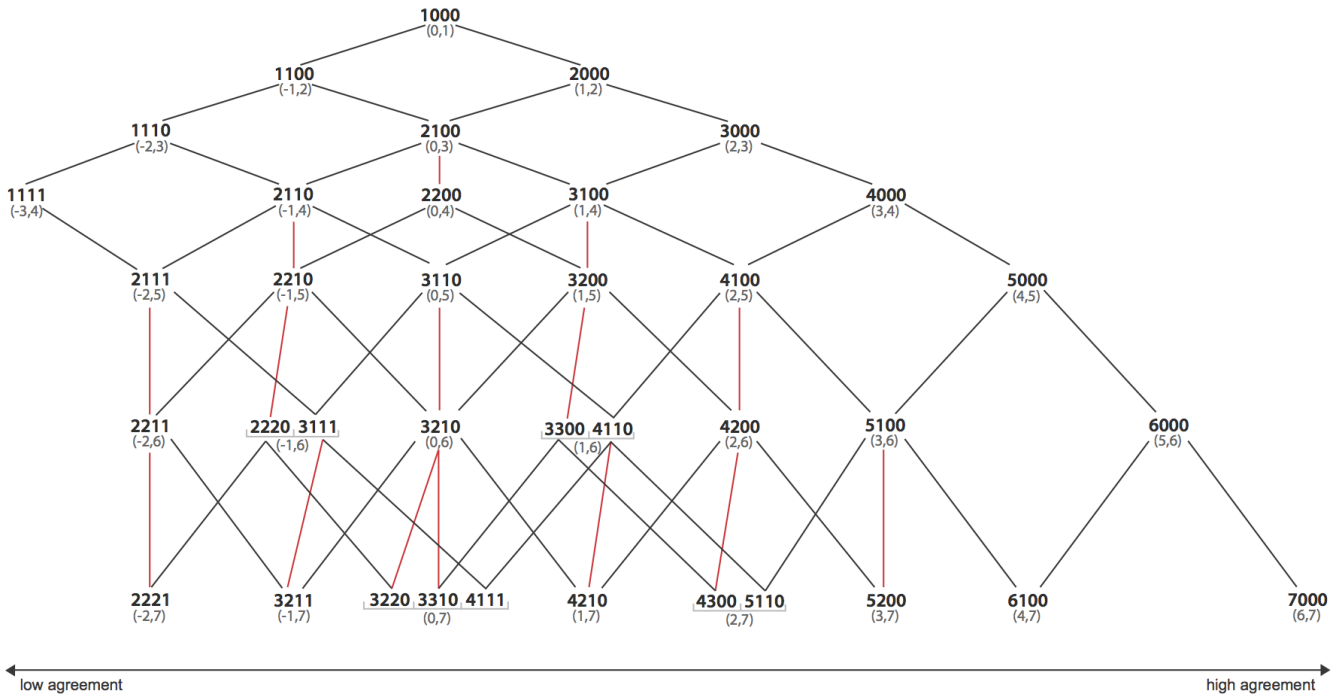

Figure 2. The ranking metric as conceptualized by the n-ary tree. Each node represents a state of agreement, and each depth level d represents all the possibilities of coding an instance with d coders. NOTE: The metric only consists of a ranking (low to high agreement) between several coding distributions as input by a participant, *e.g.*, (in order of agreement) $...\{4000\} < \{5000\} < \{6000\}...$, and so on. The tree visualization only offers a conceptual understanding into how the metric is operationalized.

involves using the right design for assigning coders to subjects that allows the use of regular statistical methods [19]. Likewise, we use a fully crossed design for displaying the data to our participants in the user study [27]. We offer participants two different representations—*(1) a table and (2) a visualization with horizontal stacked bar-charts*—to inform the participants how different instances of data are labeled by different coders. The two different types of information representation constitute our independent measure. Participants self-reported their perceived ranking of agreement (i.e., the dependent measure) of data instances that coders had labeled in a dataset. Our study used a between-subjects counterbalanced measures designed to adjust the order effect of learning from one of the representations.

Participants

We recruited 50 participants through the Amazon Mechanical Turk (MTurk) platform and paid each person $2.25 for completing the task. This compensation reflected a minimum wage payment. All participants were at least 18 years of age, had minimal experience with qualitative coding, and provided consent prior to beginning the first task.

Stimuli

We displayed information about how seven coders had labeled five data instances using four different labels in one dataset, with a total of four such datasets. Each dataset was pseudo-randomized to contain both data instances which we expected would contain low agreement and high agreement as ranked by the post-traversal and n-ary tree algorithms. Participants saw

	Coder 1	Coder 2	Coder 3	Coder 4	Coder 5	Coder 6	Coder 7
T1	A	C	B	B	C	A	D
T2	A	-	-	D	B	C	-
T3	B	A	-	B	A	B	A
T4	C	D	C	C	C	C	A
T5	C	B	B	D	-	D	A

Figure 3. A facsimile of the data representation in the user study indicating 5 data instances coded by 7 coders using 4 different codes. The empty cells indicate that the corresponding coder chose not to code that instance.

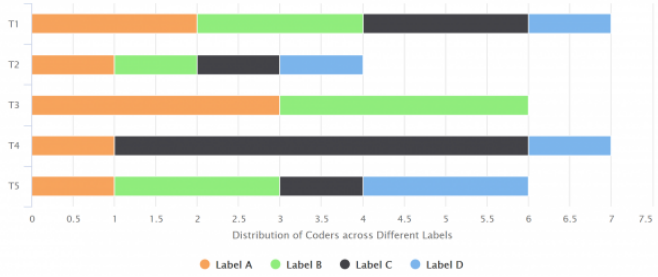

Figure 4. The visualization from user study with horizontally stacked bar-charts representing the state of agreement across the 5 data instances amongst 7 different coders. Each color represents a unique code.

this information using two representations - a table (Figure 3) and a stacked-bars visualization (Figure 4). Empty cells in the table indicated that the corresponding coder chose not to label that instance. Similarly, a length shorter than 7 in any of the stacked-bars indicated that one or more coders neglected to assign a label to that particular instance. We referred to our data instances, which were described simply as *text* to participants, as T1-T5. The labels A, B, C, D were used as agnostic codes to focus solely on perceived agreement without context and to remove potential bias from label names.

Tasks

1. *Relative ranking on a common linear scale of agreement*: Participants viewed a dataset of five data instances either in a table or a visualization and ranked them on a vertical linear scale based on their perceived amount of coding agreement. They clicked and dragged instances above or below another instance to increase or decrease the rank of the clicked instance. Rankings were numbered $1-5$ without any overlapping ranks. The rank 1 represented most agreement while 5 represented least agreement. We informed the participants that order of ranking would be immaterial between instances which they believed would have similar agreement amongst coders.

2. *Absolute ranking on separate linear scales of agreement*: Participants ranked the instances individually by dragging a *dot* across a horizontal linear scale left or right to indicate lower or higher agreement. They repeated the process five times, once for each instance in the dataset. It allowed us to confirm whether their suggested ranking in this task matched their reported ranking in the previous task.

Procedure

Participants were informed that each dataset contained five data instances which had each been coded by seven coders. Each participant received four different datasets, one at a time, and completed both ranking tasks for each set before seeing the next set. Group 1 (25 MTurk participants) received the coding information for first two data-sets A and B as a table (Figure 3), and as a visualization for the latter two sets, C and D. (Figure 4). Group 2 (25 MTurk participants) viewed the same dataset presented in reverse order, i.e., the first two datasets were visualizations while the latter two were presented as tables.

Results

Aggregated Results

On average, each participant spent about 9 minutes and 20 seconds completing the survey; minimum time was 3 minutes and 11 seconds, while maximum was 27 minutes. We recorded the participant-reported ranks for the degree of agreement within the 5 data instances in each of the 4 datasets used in the study. Similarly, we recorded their reported percentage of agreement using the slider scale. In order to compare whether the post-traversal tree metric and n-ary tree metric ranking better correlated with user perception of agreement, we averaged participant-reported responses and computed their correlations with those drawn from both the tree metrics. We ran a Spearman's Rho correlation to establish the correlation between ranks and used Pearson's coefficient for computing the correlation between the slider values and the tree-rankings. The correlation results for each dataset, task type, and data presentation format are shown in Table 1 for the post-traversal tree-based metric and in Table 2 for the n-ary tree-based metric. We refer to these correlations of averages as aggregated results. In general, we observed that the n-ary tree-based rankings strongly correlated with the user perception of agreement. More analysis of these results is in the Discussion Section.

Finer-Grained Results

Given the very strong correlation of our n-ary ranking metric with aggregated rankings and slider values, we were interested in exploring finer-grained correlations with individual participants' rankings and slider values for disagreement. Accordingly, we computed correlations for each participant's rankings and slider values with the n-ary tree ranking metric, and then determined the average correlations. We tested the ranking task data with Pearson's R. In order to capture the magnitude values represented in the slider data, we correlated slider values with the n-ary tree metric using Spearman's Rho. We then applied the Fisher Transformation [7, 17, 14, 15] to stabilize the variance of the correlation coefficients. Since the Fisher Transformation is undefined for the values of -1 and 1, we represented correlations of -1 as -0.9999 and 1 as 0.9999. The resulting correlation coefficient (R) and 95% confidence interval for each combination of dataset, task type, and data presentation format are shown in Table 3. The same table also reports the summary of correlation statistics for all datasets by task type and data presentation format using the same techniques. We discuss these results along with the aggregated results from before in a later section.

QUALITATIVE USER STUDY WITH EXPERT CODERS

To test whether qualitative researchers understand qualitatively coded data with the same proposed strategies as described by *n-ary* and *post-traversal* tree metrics, we invited 8 expert qualitative coders (4 female) to complete the same survey used in the MTurk study. Six experts were PhD students and two were undergraduate research assistants supervised by a PhD student. They all frequently engaged in qualitative research, including conducting, coding, or analyzing qualitative studies.

Procedure

Expert participants P1-P4 saw the same four sets of data as the participants from Group 1 on MTurk, where datasets A and B were displayed as tables and the latter two, C and D, were displayed as visualizations. P5-P8 saw the same sets of data as Group 2, where sets A and B were displayed as visualizations and sets C and D were displayed as tables. Upon survey completion, we asked each expert to answer a set of post-survey questions in a brief interview. The total time for each expert amounted to no more than 30 minutes, with an average time of 23 minutes. Each expert participant was compensated with a reward value of $5.00.

Interviews with experts focused on each person's thought process and strategy for deciding ranks and identifying agreement for each data instance. From the interviews, we found that experts mainly used two strategies to identify agreement: 1)

		Table		Visualization		Overall	
		R	**P value**	**R**	**P value**	**R**	**P value**
Ranking	Set A	1	0	0.9	0.037	1	0
	Set B	1	0	1	0	0.9	0.037
	Set C	0.975	0.005	0.9	0.037	0.9	0.037
	Set D	0.7	0.188	0.9	0.037	0.9	0.037
	All Sets	0.88	<0.00001	0.902	<0.00001	0.905	<0.00001
Sliders	Set A	-0.9829	0.002654	-0.9748	0.004727	-0.9867	0.001776
	Set B	-0.875	0.052046	-0.8572	0.063503	-0.8764	0.05143
	Set C	-0.9652	0.007819	-0.9605	0.009191	-0.9641	0.008155
	Set D	-0.6326	0.251679	-0.9147	0.029366	-0.842	0.073578
	All Sets	-0.851	0	-0.901	0	-0.903	0

Table 1. Aggregated correlation results for each combination of dataset, task type, and data presentation format using the post-order traversal tree metric. Overall results by dataset and task type are also shown. Each dataset was pseudo-randomized to contain both data instances with low agreement and high agreement.

		Table		Visualization		Overall	
		R	**P value**	**R**	**P value**	**R**	**P value**
Ranking	Set A	1	0	0.9	0.037	1	0
	Set B	0.975	0.005	0.975	0.05	0.975	0.005
	Set C	1	0	0.975	0.005	0.975	0.005
	Set D	0.82	0.088	0.975	0.005	0.975	0.005
	All Sets	0.933	<0.00001	0.948	<0.00001	0.959	<0.00001
Sliders	Set A	-0.9829	0.002654	-0.9748	0.004727	-0.9867	0.004727
	Set B	-0.9979	0.000107	-0.9674	0.00716	-0.9954	0.00716
	Set C	-0.9833	0.002654	-0.9861	0.001984	-0.9867	0.001984
	Set D	-0.7693	0.12855	-0.9773	0.004173	-0.9341	0.004173
	All Sets	-0.913	0.00001	-0.945	0	-0.958	0

Table 2. Aggregated correlation results for each combination of dataset, task type, and data presentation format using the n-ary tree metric. Overall results by dataset and task type are also shown. Each dataset was pseudo-randomized to contain both data instances with low agreement and high agreement.

counting the number of different labels used, and 2) counting the frequency a label was used, especially when presented as a visualization. These two main strategies align with our proposed strategies of using diversity (n-ary tree metric) and divergence (post-traversal metric) as a means of distinguishing varying levels of agreement.

Strategies to Understand Data in Tables & Visualizations

Experts generally believed that looking for agreement was a *"process of divergence and diversity"* (P4). Although only P4 explicitly used this terminology, other experts supported the point when discussing their strategies for ranking agreement.

Diversity vs Divergence

All experts reported counting the number of labels used for each data instance as a primary strategy when interpreting a table. Furthermore, all experts also reported comparing the number of different colors in each stacked bar graph of the visualization. Both of these strategies demonstrate a trend of approaching data by identifying the variety, or in other words, the diversity of a data instance. Experts believed that more labels used in the coding process conveyed more disagreement:

"The more choices there are, the more disagreement." - P5

"A 3-way split is better than a 4-way split [of colors]." - P4

Although all experts used diversity to help them decide agreement in data instances, some experts prioritized divergence over diversity. When experts consider the number of coders who used a specific label before considering the amount of variation in a data instance, they are prioritizing divergence. P1 notes, *"Most agreement is like... the one that has the longest bar of one color."* This form of thinking correlates with the post-traversal metric which prioritizes strength of disagreement before identifying the amount of diversity.

In most cases, both strategies were used. P6 described his use of diversity and divergence when explaining the rationale behind his decisions for a visualization. *"First thing I'm looking at is, there should be less colors, right? Because then there would be less labels attached. And then I'm looking at if there's a majority of a certain color."* The initial search for colors supports the idea behind diversity—he identified the number of unique colors to determine the ways in which the coders disagreed. The latter half of P6's rationale checked for divergence—he compared the amount of color there was for each label to gauge the degree to which coders disagreed. P1 also followed the same process to decide rankings: *"If it's split up three different ways, it's obviously going to be the one that has more towards one [color that has] the most inter-rater reliability."* Both of these cases demonstrated instances where experts disambiguated agreement levels in data instances by following the n-ary tree metric, which retains the complexity of qualitative coding by considering both diversity and divergence as the algorithm constructs each level of its tree.

		Table		Visualization		Overall	
		R	**Confidence Interval**	**R**	**Confidence Interval**	**R**	**Confidence Interval**
Ranking	Set A	0.965	(0.924, 0.984)	0.956	(0.905, 0.98)	0.961	(0.932, 0.977)
	Set B	0.937	(0.865, 0.971)	0.829	(0.655, 0.919)	0.896	(0.824, 0.939)
	Set C	0.841	(0.677, 0.925)	0.909	(0.807, 0.958)	0.879	(0.797, 0.929)
	Set D	0.615	(0.306, 0.806)	0.925	(0.841, 0.966)	0.824	(0.712, 0.896)
	All Sets	0.859	(0.766, 0.917)	0.896	(0.825, 0.939)	0.837	(0.731, 0.903)
Sliders	Set A	-0.977	(-0.989, -0.948)	-0.968	(-0.985, -0.93)	-0.973	(-0.984, -0.953)
	Set B	-0.942	(-0.973, -0.875)	-0.8	(-0.905, -0.604)	-0.891	(-0.936, -0.817)
	Set C	-0.806	(-0.908, -0.614)	-0.912	(-0.959, -0.814)	-0.868	(-0.923, -0.78)
	Set D	-0.511	(-0.746, -0.162)	-0.899	(-0.953, -0.788)	-0.768	(-0.86, -0.626)
	All Sets	-0.725	(-0.833, -0.563)	-0.753	(-0.851, -0.604)	-0.754	(-0.852, -0.606)

Table 3. Fine-grained average correlation results for each combination of dataset, task type, and data presentation format using the n-ary tree metric. Overall results by dataset and task type are also shown. Each dataset was pseudo-randomized to contain both data instances with low agreement and high agreement.

DISCUSSION

Post-Traversal Tree Metric vs N-ary Tree Metric

We observe higher overall R coefficient values for the n-ary tree metric in Table 2 than for the post-traversal tree metric in Table 1 considering both participant-reported ranking (0.959 vs 0.905) and slider scores (-0.958 vs -0.903) across all sets of data. The same trend is preserved across pairwise comparison of R coefficient values across individual data sets, suggesting that the n-ary tree metric better represents user perception of disagreement than the post-traversal tree metric.

The *diversity* of conflicting labels (i.e., the different labels used for coding) and the *divergence* or "strength" of a disagreement (i.e., the number of coders who applied differing labels) are important factors that help us appropriately rank collective disagreement in collaborative qualitative coding. Both the proposed rankings align with one of these two methods for thinking through disagreement– one based on divergence, and the other based on diversity. The post-traversal tree metric prioritizes divergence over diversity as the reason for more disagreement. On the other hand, the n-ary tree metric weighs diversity of labels over divergence for deciding disagreement.

Analysis suggests that the majority of the MTurk participants resonated with diversity as the stronger factor for deciding disagreement. Most experts agreed, but some expert coders verbally reasoned and associated more divergence with higher disagreement. This became especially clear when they ranked the coding distribution $\{3220\}$ to have higher agreement than $\{3310\}$. The experts who would have preferred not to rank such ambiguous instances ranked $\{4111\}$ higher than $\{3220\}$, which are ranked as equal using the n-ary tree metric. This makes us thoughtful about the possibility of a finer metric that balances divergence with diversity, and optimally ranks several states of disagreement, which otherwise are ranked as equal using the n-ary tree metric.

Portraying disagreement: Table vs. Visualization

When asked to compare personal preference and perceived difficulty of judging disagreement using a table versus a visualization, most expert coders sided with the visualization. Set D also demonstrates that MTurk participants had better judgment of resolving cases with similar degrees of disagreement when looking at the visualization. Results show that participants understood disagreement much better in visualizations. Most expert coders affirmed this, claiming that visualizations were much easier to read and comprehend. However, there were some experts who preferred tables over visualizations because tables displayed which coder assigned which label. In general, the results of the stacked-bars visualization reveal rankings aligning much more closely to user perceptions of disagreement than table-based rankings.

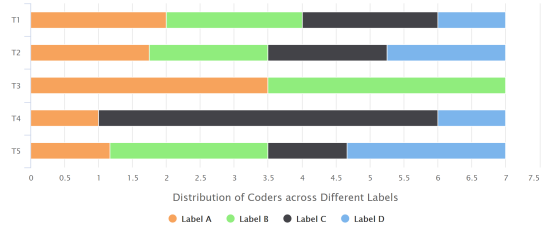

Figure 5. The normalized visualization allows easier comparison by creating equal length bars even for instances having an unequal numbers of coders.

Although the visualization in Figure 4 offered granularity of the represented data, many expert coders reported confusion due to the uneven length of bars. Some experts explained that having an uneven number of coders makes comparisons across data instances much more difficult. To resolve this issue, we suggest that future visualizations use normalized bars, which distributes coders' responses proportionally in a data instance to fit the same total bar length in all data instances. Figure 5 demonstrates a normalized view of the same data represented in Figure 4. The normalized visualization should help participants avoid the confusion of empty spaces.

Significance and applicability of proposed metric(s)

We revisit the two case studies described earlier to better understand the contribution after our user studies. Tree-based ranking metrics provide a mechanism for exploring and valuing different forms of disagreement. Rather than accepting the perspective of one authority figure—a concern that arose in Case Study I—similar types of disagreement can be identified using these metrics. The metrics also support identification of

systematic disagreements, which have been shown to reveal meaningful trends in data [2, 11, 21].

When coding a data set, researchers tend to disagree on several instances, too many to spend time discussing face-to-face in a group. For situations with massive amounts of data to be coded—such as the research presented in Case Study II—the tree-based metric reduces human effort in evaluating disagreement consistently. It sorts all the ambiguous instances, so that the *most ambiguous* could be shown to the coders first, then gradually decreasing in ambiguity. That way if they run out of time, they will have worked on the most challenging and/or confusing tweets first. As any qualitative researcher knows, one of the challenges in this type of work is dealing with massive amounts of data, and in trying to figure out what is most relevant to present in a memo or study. Although visual analysis for identifying ambiguous instances [22] is useful, our approach augments human capacity by making it easier to identify contentious instances.

The tree-based metrics use coding distribution to evaluate disagreement, and offer systematically different coder perspectives even for vastly different data. Our metric can be valuable to qualitative researchers who are not computer scientists when used within a tool like *Aeonium* [10]. Such an offering is immensely useful to build more sophisticated tools to aid qualitative researchers in the following ways:

1. *To augment researcher-effort for sorting out a coding scheme:* Researchers use their own analytical skills for sorting the overall coding (or its subset). This process is extremely demanding of human efforts. Some of the researchers often resort to the use of visualizations to aid their efforts. The tree-based metric can be used to build more effective visualizations based upon its specific rank-ordering of disagreement to highlight instances that need researcher attention.

2. *To learn across different coding patterns and datasets:* At times, the coding patterns across subsets in a dataset, or across different datasets, are similar. However, a unique dominant code in the subset, or use of non-identical codes across datasets obscure the similarity. Our approach offers researchers an opportunity to identify such similarity by using a conceptual framework that is independent of the coding schemes or the inherent meanings of the codes.

3. *To capture researcher-specific meaning of disagreement:* Humans have subjective alignments when they disambiguate. Consider a case where one qualitative researcher is not interested in a majority vote, but in the split across the codes on a data point. At the same time, another researcher may want to inspect the coding scheme through another interpretation of what matters more (as illustrated in the second user study). At present, we do not have standardized metrics that allow a team to align their understanding of what disagreement means to them. Our approach offers a computational lens to disagreement—as understood by the researcher—and supports the diverse perspectives.

The qualitative coding process often involves creating new codes as researchers find fit. At present, our approach does not support creation of codes as new exemplars arise. However, our approach (with minor adjustments) can work well when coders assign multiple labels per data instance. For example, in the worst case scenario of n coders assigning all the m labels to a data instance, a state of disagreement will be represented using a tuple of length mXn.

The metric-suggested rankings of disagreement may not lie along the dimensions that match any known theory of judgment and decision-making. We believe that aligning our metrics within an existing theory of decision-making may not necessarily be a good idea as these metrics have often been based on the premise of modeling purely rational behavior, which recent research in economics and psychology has shown to be inaccurate [8]. In our paper, we have produced sufficient empirical evidence in support of the metric. While our approach does not promise a specific methodology to immediately improve the qualitative coding process, we provide a robust conceptual framework useful to devise several methodological processes to suit the personalized needs of qualitative researchers.

CONCLUSION

Human intuition for comparing across different states of disagreement can be severely challenged with increasing amounts of data to be coded and limited available resources for coding it. This is further complicated by coder bias when dealing with the complexity of disagreement [26]. However, a state of (dis)agreement is independent of the labels used for coding data instances. The paper presents a conceptual understanding of collaborative disagreement that remains indifferent to the coding context and groups of coders irrespective of their discipline. We use this conceptual formulation to offer tree-based ranking metrics that allow coders to order different states of coding disagreements to discern ambiguity. Our proposed approach of dealing with disagreement treats all the labels uniformly, and remains unchanged with new coding schemes provided the number of unique labels is preserved. This agnostic property offers qualitative coders an opportunity to easily analyze disagreement, resolve minor disputes, single out irregular instances, and help improve coding of the data. With such properties, the metrics successfully represent ambiguous instances such that they match the coder's perceptions of disagreement.

REFERENCES

1. David Armstrong, Ann Gosling, John Weinman, and Theresa Marteau. 1997. The place of inter-rater reliability in qualitative research: an empirical study. *Sociology* 31, 3 (1997), 597–606.

2. Lora Aroyo and Chris Welty. 2015. Truth is a lie: Crowd truth and the seven myths of human annotation. *AI Magazine* 36, 1 (2015), 15–24.

3. Laila Burla, Birte Knierim, Jurgen Barth, Katharina Liewald, Margreet Duetz, and Thomas Abel. 2008. From text to codings: intercoder reliability assessment in qualitative content analysis. *Nursing research* 57, 2 (2008), 113–117.

4. John L Campbell, Charles Quincy, Jordan Osserman, and Ove K Pedersen. 2013. Coding in-depth semistructured interviews: Problems of unitization and intercoder reliability and agreement. *Sociological Methods & Research* 42, 3 (2013), 294–320.
5. Joseph Chee Chang, Saleema Amershi, and Ece Kamar. 2017. Revolt: Collaborative Crowdsourcing for Labeling Machine Learning Datasets. In *CHI 2017*.
6. Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement* 20, 1 (1960), 37–46.
7. David M Corey, William P Dunlap, and Michael J Burke. 1998. Averaging correlations: Expected values and bias in combined Pearson rs and Fisher's z transformations. *The Journal of general psychology* 125, 3 (1998), 245–261.
8. Benedetto De Martino, Dharshan Kumaran, Ben Seymour, and Raymond J. Dolan. 2006. Frames, Biases, and Rational Decision-Making in the Human Brain. *Science* 313, 5787 (2006), 684–687. `http://science.sciencemag.org/content/313/5787/684`
9. Ryan Drapeau, Lydia B Chilton, Jonathan Bragg, and Daniel S Weld. 2016. Microtalk: Using argumentation to improve crowdsourcing accuracy. In *Fourth AAAI Conference on Human Computation and Crowdsourcing*.
10. M. Drouhard, N. C. Chen, J. Suh, R. Kocielnik, V. Pena-Araya, K. Cen, Xiangyi Zheng, and C. R. Aragon. 2017. Aeonium: Visual analytics to support collaborative qualitative coding. In *2017 IEEE Pacific Visualization Symposium (PacificVis)*. 220–229.
11. Anca Dumitrache. 2015. Crowdsourcing Disagreement for Collecting Semantic Annotation. In *European Semantic Web Conference*. Springer, 701–710.
12. Nicholas Epley and Thomas Gilovich. 2006. The anchoring-and-adjustment heuristic Why the adjustments are insufficient. *Psychological science* 17, 4 (2006), 311–318.
13. Frank Fischer, Johannes Bruhn, Cornelia Gräsel, and Heinz Mandl. 2002. Fostering collaborative knowledge construction with visualization tools. *Learning and Instruction* 12, 2 (2002), 213–232.
14. Ronald A Fisher. 1915. Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population. *Biometrika* 10, 4 (1915), 507–521.
15. Ronald A Fisher. 1921. On the probable error of a coefficient of correlation deduced from a small sample. *Metron* 1 (1921), 3–32.
16. D Randy Garrison, Martha Cleveland-Innes, Marguerite Koole, and James Kappelman. 2006. Revisiting methodological issues in transcript analysis: Negotiated coding and reliability. *The Internet and Higher Education* 9, 1 (2006), 1–8.
17. Richard L Gorsuch and Curtis S Lehmann. 2010. Correlation coefficients: Mean bias and confidence interval distortions. *Journal of Methods and Measurement in the Social Sciences* 1, 2 (2010), 52–65.
18. Kilem Li Gwet. 2008. Computing inter-rater reliability and its variance in the presence of high agreement. *Brit. J. Math. Statist. Psych.* 61, 1 (2008), 29–48.
19. Kevin A Hallgren. 2012. Computing inter-rater reliability for observational data: an overview and tutorial. *Tutorials in quantitative methods for psychology* 8, 1 (2012), 23.
20. Daniel J Hruschka, Deborah Schwartz, Daphne Cobb St John, Erin Picone-Decaro, Richard A Jenkins, and James W Carey. 2004. Reliability in coding open-ended data: Lessons learned from HIV behavioral research. *Field Methods* 16, 3 (2004), 307–331.
21. Sanjay Kairam and Jeffrey Heer. 2016. Parting Crowds: Characterizing Divergent Interpretations in Crowdsourced Annotation Tasks. In *CSCW 2016*.
22. Katie Kuksenok, Michael Brooks, John J Robinson, Daniel Perry, Megan K Torkildson, and Cecilia Aragon. 2012. Automating large-scale annotation for analysis of social media content. In *IEEE Workshop on Interactive Visual Text Analytics for Analysis of Social Media*.
23. Walter S Lasecki, Mitchell Gordon, Danai Koutra, Malte F Jung, Steven P Dow, and Jeffrey P Bigham. 2014. Glance: Rapidly coding behavioral video with the crowd. In *UIST 2014*.
24. Margaret D LeCompte. 2000. Analyzing Qualitative Data. *Theory into practice* 39, 3 (2000), 146–154.
25. Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica* 22, 3 (2012), 276–282.
26. Slava Mikhaylov, Michael Laver, and Kenneth R Benoit. 2012. Coder reliability and misclassification in the human coding of party manifestos. *Political Analysis* 20, 1 (2012), 78–91.
27. Dan J Putka, Huy Le, Rodney A McCloy, and Tirso Diaz. 2008. Ill-structured measurement designs in organizational research: Implications for estimating interrater reliability. *Journal of Applied Psychology* 93, 5 (2008), 959.
28. Johnny Saldaña. 2015. *The Coding Manual for Qualitative Researchers*. Sage.
29. Anselm L Strauss. 1987. *Qualitative Analysis for Social Scientists*. Cambridge University Press.
30. Jasy Liew Suet Yan, Nancy McCracken, Shichun Zhou, and Kevin Crowston. 2014. Optimizing features in active machine learning for complex qualitative content analysis. *ACL 2014* 44 (2014).