Participant Self-Coding as Workflow Signal: Codebook Repair and the Concreteness Gradient in Participatory Qualitative Analysis

ANONYMOUS AUTHOR(S)

Participatory qualitative-analysis systems need to decide when participant self-coding can be trusted, when researcher review is needed, and when the codebook needs repair. We study this workflow problem in a deployment with 879 participants who wrote open-text explanations, self-coded with a 306-code checklist, proposed missing codes, and were compared with researcher consensus on a matched subset. Researchers retained 22.4% of participant selections: when participants selected a code, that response–code pair appeared in the final researcher consensus about one time in five. Disagreement was not uniform across codes. Rejection rates also varied widely across individual codes: averaged within the third of codes with the lowest rejection, researchers rejected 67% of participant selections, and averaged within the third with the highest rejection, they rejected 97%. Because a code's retention rate is one minus its rejection rate, this code-level spread is what produces the difference in retention between code groups described below. A separate qualitative inspection of the most- and least-divergent codes shows that the highest-disagreement codes describe experiential reactions while the lowest-disagreement codes describe more text-grounded features. Participant explanations helped explain this contrast. When researchers did not retain participant selections, those selections often had very little word overlap with the code's description. To measure this, we compared the content words in a participant's explanation with the content words in the selected code description, after removing common stop words. Higher overlap means the participant used wording closer to the code; it is evidence of textual support, not proof that the participant and researcher interpreted the case the same way. These rejections are not automatically participant errors. In a sampled rule-based follow-up of 50 rejected selections that had basic word overlap, 23 met a stricter deterministic textual-support criterion (46.0%, 95% CI [32.0, 60.0]). A retrospective routing simulation, evaluated under held-out k=5 response-fold cross-validation, preserved 76.2% [74.7, 77.8] of selections researchers retained while removing 53.7% [52.3, 54.9] of participant selections from review (in-sample numbers, 79.1% and 53.8%, sit on the optimistic side of these intervals). Selective adjudication means routing only disagreement-prone selections to researcher review, and this result should be treated as a design hypothesis rather than proven live-workflow performance. A separate analysis of 87 participant-originated code suggestions shows that most were near-duplicates of existing codes, suggesting that suggestion channels mainly reveal phrasing, example, and definition gaps. Together, the findings show how participatory coding tools can preserve participant voice through trust, routing, and repair instead of treating participant labels as either ground truth or noise.

CCS Concepts: • **Human-centered computing → Empirical studies in collaborative and social computing**; *Collaborative and social computing systems and tools*; *Empirical studies in HCI*.

Additional Key Words and Phrases: participatory sensemaking, qualitative coding, codebook design, participant–researcher divergence, concreteness gradient

ACM Reference Format:
Anonymous Author(s). 2023. Participant Self-Coding as Workflow Signal: Codebook Repair and the Concreteness Gradient in Participatory Qualitative Analysis. 1, 1 (November 2023), 26 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Participatory qualitative analysis is collaborative work among participants, researchers, tools, and governance rules. The interface determines whether participant voice enters analysis as evidence, as noise, or as a prompt to repair the codebook. When participants code their own qualitative responses, tools therefore need more than an agreement score. They need to know when participant

selections are candidates for provisional trust, when they require adjudication, and when participant language reveals that the codebook is hard to use.

This paper shows that participant–researcher disagreement follows a predictable structure. We call this structure the *concreteness gradient*. By *concreteness gradient*, we mean the empirical pattern that participants and researchers agree more on codes describing observable, text-grounded features and disagree more on codes describing abstract or experiential reactions. In a participatory sensemaking deployment, 879 participants self-coded their responses using a 306-code checklist while researchers independently coded the same data. We call a participant selection *retained* when researchers keep that response–code pair in the final consensus. Retention rate asks a practical question: when participants selected this code, how often did researchers keep that selection in the final consensus? Participant selections had low agreement with the final researcher consensus (precision = 0.224, where precision is the share of participant selections that researchers retained), but the disagreement was not random. Participants and researchers agreed more on concrete, text-grounded codes and diverged more on abstract, experiential codes. Here, *concrete codes* describe features verifiable from the passage text, and *abstract codes* describe internal experiences not directly verifiable from the passage. Code-level FP rates spanned a wide range, from a mean of 0.67 in the lowest tertile to 0.97 in the highest tertile (Section 5.5); because retention rate is mathematically 1 − FP rate for a single code, this is the same spread expressed from the retention side rather than an independent finding. A separate qualitative inspection of the highest- and lowest-divergence codes (Section 5.4) shows that the highest-FP codes describe experiential reactions while the lowest-FP codes describe more text-grounded features. We treat code-level FP rates as a workflow signal because they tell us, for each code, how often participant labels were retained in this dataset and therefore which codes might be candidates for provisional trust versus researcher review.

This structure makes disagreement usable for workflow design. Some participant selections are candidates for provisional trust; others need researcher review or richer participant evidence. By *selective adjudication*, we mean routing only the participant selections that historically generate disagreement to researcher review, while accepting the rest as-is. A retrospective simulation shows that selectively reviewing codes based on their historical disagreement rates preserved 76.2% [74.7, 77.8] of selections researchers ultimately kept while reducing reviewed selections by 53.7% [52.3, 54.9] under held-out k=5 response-fold cross-validation (in-sample 79.1% and 53.8%; Section 5.6). We also checked whether rejected participant selections still contained useful written evidence. Participants explained their code choices in their own words. To check whether an explanation pointed to the same idea as the selected code, we compared the content words in the explanation with the content words in the code text, after removing common stop words. We report this match as Jaccard overlap: shared content words divided by all unique content words across the two texts. Higher overlap means the participant used wording closer to the code; it is evidence of textual support, not proof that the participant and researcher interpreted the case the same way. In a sample of 50 rejected selections that had basic word overlap with the code description, 23 met a stricter deterministic textual-support criterion (46.0% [32.0, 60.0]; Section 5.8).

Common qualitative coding tools—NVivo, Dedoose, and AI-augmented platforms [13, 32]—are built for trained analysts, not for participants. Participants generate the data. Researchers interpret it. This split works, but it means participants cannot push back on the labels researchers put on their words [3, 23]. Participatory design traditions call for returning analytic authority to participants [3, 21], but participatory coding has stayed small-scale.

We deployed an interactive system that bridges this gap. Participants contribute self-coded labels and suggest vocabulary repairs through a checklist interface. Researchers review and integrate those codes through a shared governance workflow (Section 3). Each respondent provided forced-choice

judgments, open-text justifications, and optional self-coding. Participants could also propose new codes when the existing taxonomy did not capture their experience.

A companion study [1] uses the same deployment, participants, and checklist apparatus to ask a different question: whether hybrid human–LLM coding preserves participant-endorsed selections. The deployment apparatus, recruitment channels, screen-share session format, online consent procedure, and minimal-risk exempt IRB determination are inherited from that companion deployment and are not re-derived here. That paper audits LLM alignment against participant and researcher judgments. This paper instead analyzes participant-authored codebook repairs and participant–researcher divergence on the same responses. The answer—that divergence follows a concreteness gradient—has direct design implications for participatory tools. The two papers therefore share collection context and materials, but they differ in research questions, analytic comparisons, and contribution claims. Relative to participant-in-the-loop survey systems and qualitative-AI governance work, the contribution here is not LLM evaluation; it is an analysis of participant–researcher divergence as a workflow-design signal [6, 20, 27, 30, 37].

We address two research questions, ordered by their role in the CSCW workflow argument. We begin with divergence because it motivates the routing problem, then turn to codebook repair:

(1) **Participant–Researcher Divergence (RQ2):** Where do participant and researcher interpretations diverge, and what does that pattern imply for workflow design? A code-level analysis of 338 matched responses reveals the concreteness gradient. We supplement that descriptive result with a small retrospective model comparison, a selective-adjudication simulation, and a sampled check of rejected selections that were flagged by text overlap.
(2) **Codebook Repair (RQ1):** How do participants repair expert-authored codebooks? We analyzed 87 participant-originated code suggestions. Most (52.9%) were near-duplicates of existing codes, exposing phrasing gaps rather than missing themes.

Our contributions are:

(1) We document the *concreteness gradient*: abstract codes produce high participant–researcher divergence; concrete codes produce lower divergence. This reframes disagreement as a structured signal rather than uniform noise.
(2) We characterize how participants repair expert-authored codebooks at scale (879 participants, 306 codes). Most suggestions (52.9%) are near-duplicates that expose phrasing gaps, not missing themes.
(3) We test selective adjudication retrospectively. Under held-out k=5 response-fold cross-validation, a triage policy based on divergence profiles—code-level summaries of historical rejection rates and visible textual support—reduces the review queue by 53.7% [52.3, 54.9] while preserving 76.2% [74.7, 77.8] of selections researchers ultimately kept (held-out flagging precision = 0.885 [0.876, 0.896]; in-sample numbers, 53.8% / 79.1% / 0.899, sit on the optimistic side of these intervals). A separate sample of 50 text-flagged rejected selections finds that 23 meet stricter textual-support criteria (46.0% [32.0, 60.0]).
(4) We propose four design hypotheses for trust-route-repair participatory coding tools. Design Hypothesis 4 (selective adjudication routing) receives initial retrospective support from the triage simulation, the open-text evidence analysis, and the sampled rule-based check of flagged rejected selections.

These findings lead to four design hypotheses (Section 6.4): sequence open text before structured tagging (DH1), preserve both input surfaces (DH2), design suggestion channels as codebook maintenance tools (DH3), and route adjudication selectively using divergence profiles (DH4). Of the four, DH4 has the strongest evidence in this dataset, though that evidence is still retrospective. The other three are motivated by the data but not experimentally tested.

2 Related Work

2.1 Participant Voice and System Constraints

Researchers and participants have different relationships to qualitative data. Researchers interpret it; participants lived it. This tension is a longstanding concern in methodology [3, 21]. Participatory design traditions emphasize co-creation [8], but data analysis often stays small-scale [22]. CSCW work argues that analysis should be a back-and-forth between researchers and communities, not a one-way extraction of meaning [9, 16, 23].

Scaling participation introduces new challenges. The format of elicitation shapes how people think. Response alternatives act as an informational frame [31]. Participants often settle for a "good enough" answer—what Krosnick [19] calls satisficing. Glazier et al. [15] asked survey respondents to code their own answers. They found moderate agreement with researchers (kappa = 0.46) on a small set of poverty-related categories. Lei et al.'s Dynamic Surveys [20] also keeps participants in the loop, but during survey interaction: participants respond to LLM-supported follow-ups and reflect on emergent clusters while data are being collected. Our study begins after elicitation. We examine a much larger codebook (306 codes) and ask where participants and researchers diverge when coding the same completed responses, where the gap is wider (precision = 0.224; Section 5). Our interface separates checklists and open text so we can study each mode independently and translate that divergence into workflow guidance.

2.2 Collaborative Coding Systems

Qualitative data analysis software (CAQDAS) organizes researcher coding around hierarchical codebooks [35]. AI-augmented systems add scaffolding. CollabCoder [13] generates initial code suggestions with LLMs. DeTAILS [32] supports iterative rationale disclosure. PaTAT [14] enables interactive rule synthesis. Others explore AI mediation [12], rationale extraction [24], and LLM-generated visualizations [17]. Plank [26] argues that annotator disagreement signals genuine ambiguity, not noise. Qualitative-AI governance work sharpens the distinction. Schroeder et al. [30] document tensions in researcher use of LLMs, Carlson and Burbano [6] and Prahl [27] specify oversight guardrails, and Ye et al. [37] design deliberation support for researcher teams. These interventions improve researcher-facing governance, but they still assume the primary analytic loop runs through trained researchers. None of these systems gives participants a built-in way to self-code the same responses that researchers later adjudicate. Our work shows empirically how participant-generated vocabulary and participant–researcher divergence move through collaborative governance—and where workflow design should react differently.

2.3 Annotator Disagreement and Perspectivist Frameworks

A growing body of work challenges the assumption that annotator disagreement is noise. Basile et al. [2] argue that evaluation should preserve disagreement rather than collapse labels into one ground truth. Uma et al. [33] survey machine-learning methods that learn from disagreement instead of discarding it. Cabitza et al. [5] call this a *perspectivist turn*. When annotators disagree, they may hold genuinely different viewpoints rather than making mistakes.

Schaekermann et al. [29] draw a useful distinction within CSCW. Some disagreements are *resolvable*: deliberation can clarify them. Others are *irresolvable*: they reflect genuinely different but valid views. Structured deliberation only helps with the first kind. Chinh et al. [7] compare four strategies for resolving coding disagreements. They find that effectiveness depends on the source of disagreement. Khadar et al. [18] use Socratic LLMs as deliberation partners. Structured dialogue helps annotators consider alternate perspectives without synchronous coordination. Fricker's [10]

concept of *testimonial injustice* is also relevant: when participant knowledge is routinely dismissed as error, the dismissal may reflect a credibility discount rather than a genuine coding mistake.

Perspectivist surveys consolidate this shift. Xu and Jurgens [36] review over 120 NLP studies that model disagreement rather than suppress it. Wan et al. [34] show that annotator disagreement often carries interpretive insights that standard pipelines discard.

These frameworks apply directly to participatory coding. When participants code from personal experience and researchers code from what the text says, the disagreement may reflect two valid viewpoints rather than a mistake. Labeling participant selections that diverge from researcher consensus as "false positives" adopts a researcher-centric frame. From a perspectivist standpoint [5], these are valid annotations from a different viewpoint. Whether this gap constitutes testimonial injustice [10] depends on whether tools treat researcher consensus as the only correct reading or as one reading among several. Understanding the *structure* of this divergence is what lets us tell the two apart. Some disagreements can be resolved through better definitions. Others reflect genuinely different perspectives.

3 Study Context and Apparatus

We designed and deployed an interactive platform for participatory qualitative sensemaking at scale. The system serves two groups: participants who generate and self-code their data, and researchers who review and integrate those codes.

3.1 Participant Interface: Progressive Disclosure

The participant workflow (Figure 1) uses progressive disclosure across three stages. In **Step 1 (Expressive Authoring)**, the interface asks for a forced-choice judgment and a plain-text rationale. Only after the participant submits does **Step 2 (Structured Tagging)** appear. Here, participants self-code their response using a grouped codebook checklist. This ordering prevents the structured taxonomy from framing the initial written response. Participants can also propose new codes when the existing taxonomy does not capture their experience. In **Step 3 (Reflective Elicitation)**, participants report which input method they prefer and explain why.

3.2 Interactive Codebook and Suggestion Channel

The system presents 306 unique codes across 20 thematic groups. The interface randomizes the order of themes and codes for each session to reduce primacy effects. The suggestion channel works as a feedback loop. Participants enter free-text missing codes at the bottom of each theme group. This turns participant disagreement into actionable data for codebook improvement.

3.3 Researcher Adjudication Setup

On the researcher side, a dedicated coding environment presents each response for adjudication. The procedure hides the participant's self-selected codes during the first pass. This keeps the two sets of codes independent so they can be compared directly. A shared governance queue handles approval of both participant-originated and researcher-originated candidate codes.

3.4 Deployment Context

We deployed the system as part of a remote, multi-school online learning community whose intern cohorts and instructor-channel participants completed a recorded screen-share session that included the participatory-coding task. Participation was voluntary, uncompensated, and recruited across multiple channels (an online crowd-work platform, intern cohorts, and an instructor channel); admissions and instructor reviewers were blinded to condition assignments, and lower task performance could not adversely affect admissions or course outcomes. A total of 879 participants

completed two passage-comparison prompts, producing 1,758 response rows. The deployment yielded participant selections, proposed codes, and reflective preference logs. The qualitative-coding deployment was reviewed by the authors' institutional review board and received a minimal-risk exempt determination (protocol number anonymized for review); informed consent was obtained online at the start of the first session via a written consent document covering session format, voluntariness, withdrawal, the absence of compensation, the screen-sharing requirement, and the data-handling policy.

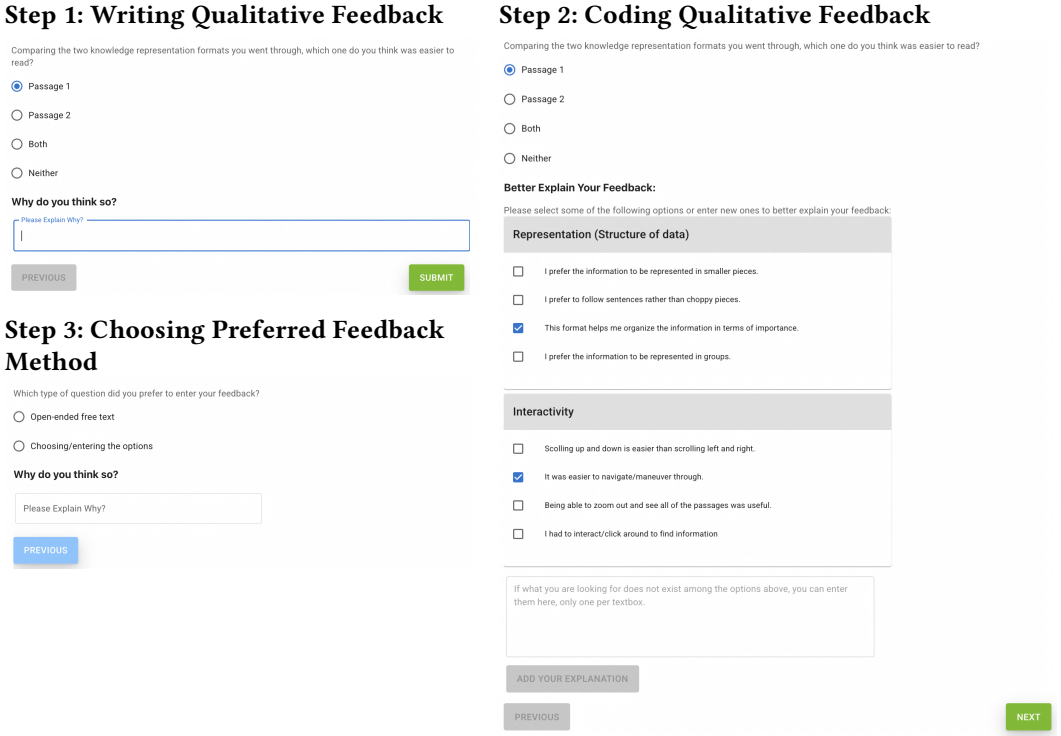

Fig. 1: Participant-facing workflow. **Step 1** (top left): forced-choice judgment and open-text justification. **Step 2** (right): self-coding checklist with suggestion channel. **Step 3** (bottom left): preferred input method and explanation.

3.5 Analytic Universes and Alignment

Tables 1 and 2 show the key subsets and summarize the alignment between participants and researchers. The full deployment produced 1,758 responses from 879 participants. Most analyses use the *matched subset*: 338 responses from 260 participants where both raw participant self-coding and fully adjudicated researcher coding exist. This subset allows us to directly compare participant selections against researcher consensus. A broader *researcher-only subset* contains 838 adjudicated responses. Supplementary Table S1 names each subset so that every claim rests on an explicit denominator.

Within the matched subset (Table 2), participants applied a total of 3,207 codes. Researchers ultimately retained 719 of these selections, resulting in a selection precision of 0.224 [0.205, 0.245]. Participants also captured roughly a third of the overall researcher consensus (recall = 0.326

Table 1: Data subsets and response volumes. The table distinguishes the full deployment size from the matched overlap (where both participant and researcher coding exist) and the broader adjudicated researcher set. We focus primarily on the matched subset to compare coding behavior.

Metric	Value
Participants	879
Open-text responses (Q1/Q2)	1758
Responses with at least 1 active participant selection	1660
Active checklist codes	306
Matched participant-researcher responses	338
Participants represented in matched overlap	260
Broader lineage-complete adjudicated responses	838
Human researchers contributing labels	34
Human coders per broader adjudicated response	median 3 [IQR 2, 3]

Table 2: Baseline participant–researcher alignment on the matched subset (n = 338 responses). Out of 3,207 total codes applied by participants, researchers retained 719 (22.4%). Although the overall precision and recall are low, participants successfully surfaced 55 of the 61 unique codes present in the final researcher consensus.

Metric	Value
Consensus-positive pairs on matched subset	2208
Raw participant selections on matched subset	3207
Overlapping participant-researcher pairs	719
Selection precision	0.224 [0.205, 0.245]
Selection recall	0.326 [0.291, 0.362]
Responses with at least 1 overlapping participant code	264
Supported codes per response when any overlap exists	2.72
Consensus-used codes surfaced by participants	55 / 61 (0.902)

[0.291, 0.362]). Here, recall is the share of researcher-retained response–code pairs that participants surfaced. Despite this low retention rate, participants managed to surface 90.2% (55/61) of the unique codes that researchers eventually used in consensus. This baseline divergence highlights the need to understand where and why participant and researcher interpretations diverge.

4 RQ1: How Do Participants Repair Expert-Authored Codebooks?

The interface let participants propose new codes whenever the available checklist felt insufficient. Over the course of the deployment, participants submitted 87 distinct suggestions, and approved codes expanded the cumulative checklist to its final size of 306 codes.

4.1 Categorization Method

We sorted each suggestion into one of four types:

- *Near-duplicate*: overlaps with existing wording.
- *Meta-comment*: about the study or task, not the passage.
- *Idiosyncratic*: tied to one person's experience, unlikely to generalize.
- *Accepted*: added to the active codebook after curation.

Table 3: Disposition of participant-originated candidate codes (n = 87). Most suggestions are near-duplicates, indicating that the channel functions as targeted codebook repair.

Category	n	%
Near duplicate	46	52.9
Meta comment	20	23.0
Idiosyncratic	17	19.5
Accepted	4	4.6
Total	87	100.0

We further sorted near-duplicates by how they repair the codebook: *reframing* (same idea, different words), *specificity* (adds concrete examples), or *granularity* (finer splits within a theme).

We classified the 87 participant-originated suggestions with a transparent rule-based scheme and inspected the outputs for face validity. The scheme uses lexical and structural features such as active status, meta-commentary markers, very short placeholder text, and long response-specific explanations. We treat this analysis as a descriptive categorization of suggestion types rather than as independently coded qualitative analysis. The classifier code and the full per-suggestion categorization are in the supplementary reproducibility materials enumerated in Supplementary Section 9, and Supplementary Table S2 reports the same generated counts used here.

4.2 Results

Table 3 shows the distribution. Most suggestions (46/87, 52.9%) are near-duplicates of existing codes. Meta-comments account for 20/87 (23.0%) and idiosyncratic proposals for 17/87 (19.5%). Four codes (4.6%) remained active after curation.

Among the 46 near-duplicates, *reframing* dominates (34/46, 73.9%), followed by *specificity* (10/46, 21.7%) and *granularity* (2/46, 4.3%). Table 4 shows the breakdown. Reframing suggestions say the same thing in different words. For example, one participant wrote "Made it more approachable to study and identify core themes" instead of the existing code "The key information was easier to identify." Specificity suggestions add concrete details that researcher wording lacks. For example, "The connecting words such as 'are called,' 'because,' or 'of' which connecting two or more sections" spells out the abstract code "There were clear, explicit links between related paragraphs/nodes." Granularity repairs seek finer distinctions. For example, "the fragmented design and the need to switch back-and-forth among sections of passage two got me lost" names a specific friction rather than a general readability issue.

Idiosyncratic proposals (17/87, 19.5%) reflected personal reading needs that a general codebook cannot cover, such as "I have ADHD and I struggle with reading in general." Meta-comments (20/87, 23.0%) gave feedback on the study itself (e.g., "I prefer interfaces with less navigation...").

The four accepted codes show what lasting repairs look like: "Encourages thorough reading (paying attention to details)," "It helps me with summarizing the information," "The arrows aided in reading comprehension," and "Visual representation makes it easier to encode details to memory." Each fills a gap in the researcher vocabulary: summarization as a learning strategy, visual encoding as a memory tool, and arrow-specific comprehension support.

4.3 Interpretation

This distribution reframes what a suggestion channel is *for*. The value is not the rare new code (4.6% acceptance rate) but the steady exposure of phrasing gaps. When 52.9% of all suggestions are

Table 4: Near-duplicate subtypes (n = 46). Reframing—expressing the same idea in different words—is the dominant repair mode.

Near-duplicate subtype	n	%
Reframing	34	73.9
Specificity	10	21.7
Granularity	2	4.3
Total near-duplicates	46	100.0

near-duplicates, participants are not struggling to find relevant themes. They are struggling to find their *exact* wording.

This adds detail to what Russell et al. [28] called "data-residue reduction." Each repair type points to a different system response. Reframing gaps call for synonym detection. Specificity gaps call for example-enriched definitions. Granularity gaps call for hierarchical expansion.

5 RQ2: Where Do Participant and Researcher Interpretations Diverge?

The matched subset (338 responses from 260 participants) lets us compare directly how participants and researchers apply the same codes. The results first ask when participant self-coding agrees with researcher consensus. They then ask whether rejected selections still contain visible textual support and whether that history can guide selective adjudication. We use several terms consistently in this section. A participant selection is *retained* when researchers keep that response–code pair in the final consensus and *rejected* when they do not. Retention rate asks a practical question: when participants selected this code, how often did researchers keep that selection in the final consensus? A *true positive* (TP) is a retained participant selection. A *false positive* (FP) is a rejected participant selection, named only from the researcher-consensus perspective. The term does not mean that the participant's interpretation was illegitimate. For any single code, retention rate and FP rate are mathematically related as retention rate = 1 − FP rate; the two summaries describe the same set of selections from opposite directions. Participants also explained their code choices in their own words. To check whether an explanation pointed to the same idea as the selected code, we compared the content words in the explanation with the content words in the code text, after removing common stop words. We report this match as *Jaccard overlap*: shared content words divided by all unique content words across the two texts. Higher overlap means the participant used wording closer to the code; it is evidence of textual support, not proof that the participant and researcher interpreted the case the same way. An *overlap-screened rejection* is a rejected selection whose open-text justification still contains enough word overlap with the code text to warrant a closer check. For each code, we summarize its past disagreement with researchers in two parts. First, we count how often researchers removed participant selections of that code. Second, we check whether the participant explanations used words that matched the code text. We call this two-part summary the code's *divergence profile*. It matters because the routing simulation uses this history to decide which future participant selections should receive researcher review.

5.1 Overall Alignment

On the 338 matched responses, participants made 3,207 checklist selections. The researchers' final consensus contained 2,208 positive response–code pairs. The two sides overlapped on 719 pairs: precision = 0.224 [0.205, 0.244], recall = 0.326 [0.290, 0.363] (Table 1, right panel). Participant

Table 5: Theme-level divergence between participant self-coding and researcher consensus on the matched subset (n = 338 responses). FP rates are researcher-consensus rejection rates, not proof of participant error. Rates range from 0.73 (Presentation) to 0.91 (Representation), with bootstrap 95% CIs.

Theme	Agreed	Part. only	Total	FP rate	95% CI
Representation (Structure of data)	7	75	82	0.91	[0.85, 0.97]
Interactivity	34	194	228	0.85	[0.80, 0.90]
Learning	84	324	408	0.79	[0.75, 0.83]
Topic-specific	114	388	502	0.77	[0.73, 0.81]
Comprehension	221	734	955	0.77	[0.73, 0.80]
Readability	107	354	461	0.77	[0.73, 0.80]
Presentation (Layout/Arrangement/Visualisation)	152	419	571	0.73	[0.69, 0.77]

selections therefore had low agreement with the researcher consensus. Participants covered about one-third of what researchers endorsed.

At the response level, only 24 responses (7.1%) were fully covered. Most (240, or 71.0%) showed partial overlap. Another 56 (16.6%) had zero overlap. Eighteen (5.3%) had no participant selection at all.

Despite the low precision, participants surfaced 55 of the 61 codes that researchers retained. Coverage spans all seven themes (Supplementary Table S12). The mismatch is local, not thematic: participant self-coding covers most of the active vocabulary but applies codes to many responses where researchers do not keep them.

5.2 Code-Level Divergence Analysis

To find *where* the two sides diverge, we computed a researcher-consensus false-positive (FP) rate for each code: the share of participant selections that researchers did not retain. From the participant's perspective, rejected selections may be legitimate readings rather than errors. Throughout the paper, *high-FP*/*low-FP* name this operational divergence rate. By contrast, *abstract*/*concrete* name the qualitative character of the code. We use *concrete* for codes anchored to observable wording or layout features in the written response, and *abstract* for codes that express experiential or evaluative judgments whose support may not appear directly in the text. Table 5 shows divergence at the theme level. Table 6 lists the codes with the highest divergence rates.

Theme-level FP rates range from 0.73 [0.69, 0.77] for Presentation to 0.91 [0.85, 0.97] for Representation. At the code level, ten codes show 100% FP rates. For these codes, every participant selection was excluded from the consensus. The most-selected is "Being able to zoom out and see all of the passages was useful" (43 selections, 0 retained). Others include "It feels more concise" (34 selections) and "The representation provides more details" (24 selections). These codes describe how the passage *felt*, not what is visible in the text.

5.3 The Concreteness Gradient

The pattern is not uniform. Codes with 100% FP rates tend to describe how the passage felt ("useful," "concise," "more details"). Participants know what they experienced but may not provide specific words that support that judgment. Codes with lower FP rates describe concrete, visible features of the text.

We interpret this pattern as evidence of different evidence standards rather than uniform noise. The claim is not that every abstract-code selection is unsupported; it is that participant and researcher evidence standards diverge more for experiential labels than for observable text features.

Table 6: Top codes by researcher-consensus FP rate on the matched subset. Multiple codes have no participant selections retained in consensus, including codes that participants selected dozens of times.

Code text	Theme	Agreed	Participant only	FP rate
Being able to zoom out and see all of the passages was	Interactivity	0	43	1.00
It feels more concise.	Readability	0	34	1.00
The representation provides more details.	Comprehension	0	24	1.00
The arrows connecting to the other paragraphs helped me	Presentation(Layout/	0	22	1.00
It helps me with summarizing the information.	Learning	0	16	1.00
It helps my learning because the format stimulates me t	Learning	0	14	1.00
It is easier to find information to answer the question	Presentation(Layout/	0	14	1.00
I like that all the information was in a single page.	Presentation(Layout/	0	7	1.00
The presentation feels monotonous to read.	Comprehension	0	6	1.00
I was able to read through the passage faster.	Readability	0	5	1.00

Two patterns argue against uniform random satisficing. First, if participants endorsed codes at random, code-level FP rates would be uninformative; the qualitative inspection above shows a clear contrast in code character at the extremes. Second, a within-participant analysis (71 participants with at least 2 matched responses and at least 3 distinct code selections) yields a median coefficient of variation across each participant's code-level FP rates of 0.166 (IQR [0.149, 0.178]; Supplementary Section 6.9). A permutation null that reshuffles each participant's selections uniformly across the codebook (preserving each participant's selection count) gives a median CV of 0.143 [0.135, 0.151]. The observed median CV exceeds the permutation null in every one of 2,000 resamples, so the within-participant variability of selection-level FP rates is reliably above what uniform random selection would produce. Neither pattern rules out satisficing entirely, but together they show that over-application reflects more than uniform random selection.

5.4 Predicting Code-Level Divergence

The concreteness gradient is a qualitative pattern. Can we predict it quantitatively? The answer is cautiously yes in this sample. We built three models for the 53 codes with at least 3 participant selections. A code counted as high-divergence when its FP rate exceeded 0.85, meaning researchers rejected more than 85% of participant selections for that code. Of these 53 codes, 31 (58.5%) met that threshold. The 0.85 cutoff is an operational choice; sensitivity at 0.75 and 0.95 is reported in Supplementary Section 6.4 and the directional pattern for selection volume is stable across thresholds. Dichotomizing at 0.85 also inherits a binomial-variance artifact: a code with very few selections can mathematically reach FP $>$ 0.85 with much higher probability than a code with many selections (Supplementary Section 6.3), so any selection-volume effect on the dichotomized outcome should be read alongside the continuous-outcome regression below.

Feature importance: what carries signal in this sample. The strongest model signal comes from how widely a code is selected. Within this retrospective sample, the exploratory random forest has the highest scores among the tested models (500 trees, max depth 4, sqrt(p) features per split, seed

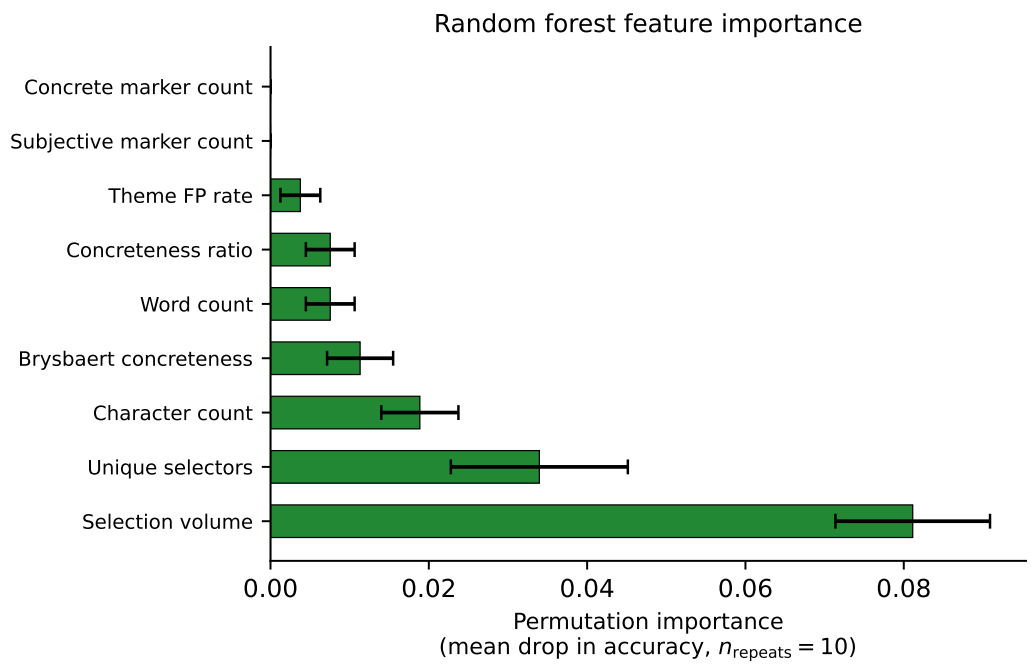

Fig. 2: Random forest permutation importance for predicting high-FP codes (mean +/- SE over 10 shuffles). Selection volume and unique selector count carry the largest importance values in this sample. Error bars show one standard error.

= 42; leave-one-out cross-validated accuracy = 0.774, F1 = 0.818). Figure 2 shows its permutation importance (mean over 10 shuffles per feature).

Two features carry the largest importance values. *Selection volume*—how often a code is selected in the matched subset—is the strongest predictor (importance = 0.0811, SE = 0.0098). *Unique selector count*—how many distinct participants chose a code—is second (0.0340, SE = 0.0112). Character count is a distant third (0.0189, SE = 0.0049), and the remaining six features each stay at 0.0113 or below.

This exploratory ranking suggests a plausible interpretation. Codes that many different people select often describe things both sides can see. Codes that few people select more often describe things only the participant felt. A portion of the dichotomized selection-volume effect is mechanically induced by the 0.85 cutoff (Supplementary Section 6.3), so the ranking should not be read as a direct measure of how many participants experienced a passage feature. Within this small retrospective sample, the ranking is more consistent with shared code uptake than with lexical surface features, but we do not treat the RF as a mechanistic model.

Qualitative inspection: top-5 vs. bottom-5 codes. The feature importance indicates *which* features carried the most signal in this retrospective sample. A qualitative comparison of the highest- and lowest-FP codes tells us *why*.

The five codes with the highest FP rates (all 1.0) describe subjective experience: “Being able to zoom out and see all of the passages was useful” (43 selections, 0 retained), “It feels more concise” (34 selections, 0 retained), “The representation provides more details” (24 selections, 0 retained), “The

Table 7: Model comparison for predicting high-FP codes. Under leave-one-out cross-validation, the random forest with nine features yields the highest accuracy and F1, modestly exceeding the logistic baselines.

Model	Features	Accuracy	Precision	Recall	F1
LR (5 features)	5	0.736	0.743	0.839	0.788
LR (9 features)	9	0.717	0.750	0.774	0.762
RF (9 features)	9	0.774	0.771	0.871	0.818

All metrics are leave-one-out cross-validated. n = 53 codes. LR = logistic regression; RF = random forest.

arrows connecting to the other paragraphs helped me with finding the information" (22 selections, 0 retained), and "It helps me with summarizing the information" (16 selections, 0 retained). Each describes how the passage felt to the participant. None points to a specific text feature a researcher could verify.

The five codes with the lowest FP rates describe more observable features: "Reading it felt more natural/comfortable" (FP = 0.509; 173 selections, 85 retained), "I prefer the vertical orientation" (FP = 0.574; 108 selections, 46 retained), "It was easier to follow the story" (FP = 0.619; 168 selections, 64 retained), "The information was better organized on the page" (FP = 0.641; 128 selections, 46 retained), and "This format helps me connect/relate ideas to each other" (FP = 0.648; 108 selections, 38 retained). These codes still describe experience, but they anchor it to something visible in the text or layout: page orientation, story structure, information organization. That anchoring is what makes them verifiable—and what makes researchers more likely to retain them.

This qualitative contrast is consistent with the quantitative feature importance. In this dataset, codes with many unique selectors are often the observable, anchorable ones. Codes with few unique selectors more often describe subjective, felt states. We treat that alignment as exploratory support for the gradient rather than as a causal account.

Model comparison. Table 7 compares all three models using leave-one-out cross-validation. A baseline logistic regression with five text-based features (word count, subjective marker count, concrete marker count, concrete ratio, selection volume) achieves LOO accuracy = 0.736. An extended logistic regression with nine features performs worse (0.717), because nine features overfit 53 observations. The random forest yields the highest leave-one-out cross-validated scores among the three models, but only modestly (accuracy = 0.774, F1 = 0.818).

The simpler logistic model points to the same feature. In the baseline logistic regression (Table 8), selection volume was the only predictor whose 95% CI excluded the null (OR = 0.18 [0.02, 0.43]). A continuous-outcome variant that regresses logit(FP rate) on the same five standardized features (weighted by per-code selection count) points the same direction without dichotomization: standardized selection volume coefficient = −0.94 [−1.39, −0.54] (Supplementary Section 6.2). The dichotomized OR therefore overstates the magnitude relative to the continuous-outcome estimate but the directional pattern survives both specifications. The other features pointed in expected directions but their confidence intervals crossed the null.

Construct-validity note. The label *concreteness gradient* rests on converging evidence, not on any single predictor. The Brysbaert et al. [4] concreteness norms enter the extended logistic regression with OR = 1.90 [0.11, 11.89]; the interval spans two orders of magnitude, so the data are uninformative about direction at this sample size. The random forest assigns only modest permutation importance to this feature (0.0113, SE = 0.0042). We therefore name the gradient

Table 8: Logistic regression predicting high-FP codes above 0.85. Selection volume is the only feature whose 95% CI excludes 1. n = 53 codes (31 high-FP, 22 low-FP). * 95% CI excludes 1.

Feature	Beta (std.)	95% CI	OR
Word count	+0.117	[-0.910, +1.429]	1.12 [0.40, 4.17]
Subjective marker count	+0.345	[-0.991, +1.915]	1.41 [0.37, 6.78]
Concrete marker count	-0.240	[-1.547, +1.103]	0.79 [0.21, 3.01]
Concreteness ratio	+0.240	[-1.406, +2.326]	1.27 [0.25, 10.24]
Selection volume	-1.707	[-4.091, -0.839]	0.18 [0.02, 0.43]*

LOO accuracy = 0.736, LOO F1 = 0.788

n = 53 codes (31 high-FP, 22 low-FP). * 95% CI excludes 1.

Table 9: Mean code-level FP rate by tertile on the matched subset. Tertiles are defined by sorting the 53 codes (each with at least 3 selections) by per-code FP rate. The spread across tertiles is the descriptive context that motivates the routing simulation in Section 5.6; it is not an independent test, since per-code retention is mathematically 1 − FP rate.

FP-rate tertile	Codes	Selections	Mean FP rate	95% CI
Low FP	17	1725	0.6701	[0.6417, 0.6962]
Mid FP	18	1025	0.8683	[0.8438, 0.8900]
High FP	18	455	0.9670	[0.9482, 0.9838]

Tertiles defined on the in-sample per-code FP rate. Per-code retention is 1 − FP rate, so this is the FP spread, not an independent test. Bootstrap 95% CIs: 2,000 resamples, seed = 42.

for the qualitative pattern—the shift from more concrete, text-grounded codes to more abstract, experiential codes along which FP rates rise—not for the psycholinguistic metric. Three lines of evidence support it: the qualitative top-vs-bottom contrast, the open-text evidence pattern across tertiles (Section 5.7), and the sampled rule-based check of flagged rejected selections (Section 5.8); the wide spread of code-level FP rates across tertiles is the descriptive context within which those tests are read, not an additional independent test. The predictive model is auxiliary because it is retrospective and limited to 53 codes.

5.5 Variation in Code-Level FP Rates

The predictive model offers a retrospective signal about *which* codes diverge. Before turning to a workflow simulation, we describe how widely the per-code FP rate varies, because that spread is what would make any triage policy worth considering in the first place. We split the 53 codes with at least 3 selections into three groups by per-code FP rate and summarize the spread across groups. Recall that for any single code, retention rate equals 1 − FP rate, so the per-tertile retention reported in Table 10 is the same information rephrased from the participant-acceptance side; we report both summaries because each is useful in different design discussions, and we treat them as one observation about the code-level FP-rate distribution rather than two independent findings.

Tables 9 and 10 and Figure 3 show the spread. Mean code-level FP rate ranges from 0.67 [0.64, 0.70] in the low-FP tertile to 0.97 [0.95, 0.98] in the high-FP tertile. The same information from the participant-acceptance side gives retention rates of 0.330 [0.302, 0.358], 0.132 [0.110, 0.156], and 0.033 [0.016, 0.052] across the three tertiles. We do not treat the cross-tertile difference as an

Table 10: Per-tertile retention rate (the same quantity as Table 9 expressed from the participant-acceptance side) on the matched subset. The wide spread of code-level FP rates is the dataset feature that motivates the triage simulation in Section 5.6; it is not an independent test of the concreteness gradient, since per-code retention is mathematically 1 − FP rate.

FP-rate tertile	Codes	Selections	Retention	95% CI
Low FP ([0.509, 0.779])	17	1725	0.330	[0.302, 0.358]
Mid FP ([0.787, 0.925])	18	1025	0.132	[0.110, 0.156]
High FP ([0.926, 1.000])	18	455	0.033	[0.016, 0.052]
Retention difference (Low - High): 0.297 [0.263, 0.331]				

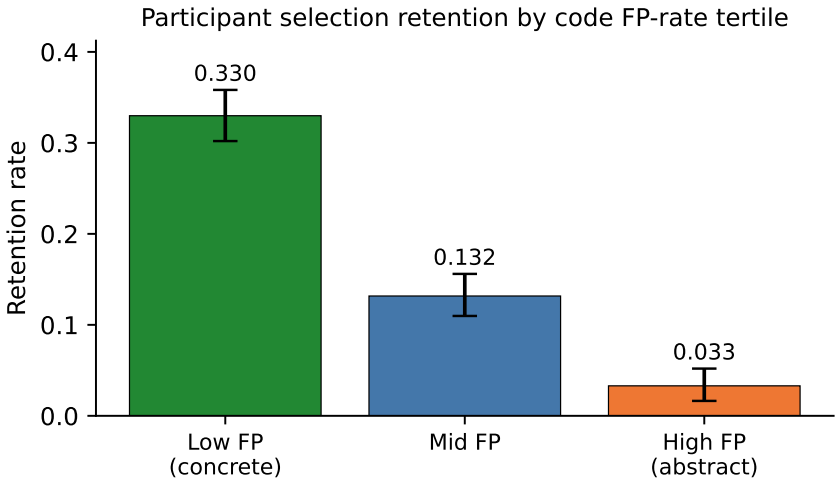

Fig. 3: Per-tertile retention rate (1 − FP rate) on the matched subset. Error bars show bootstrap 95% CIs. The descriptive spread is what makes the triage simulation in Section 5.6 worth running; it is not an independent test of the gradient.

independent test because the per-tertile retention rate is the per-tertile FP rate restated from the opposite direction; the substantive workflow test of whether this code-level spread transfers to held-out responses is the simulation in Section 5.6.

We also asked whether passage-specific references in open text (e.g., "passage 1," "arrow") predict retention at the response level. They did not (0.227 vs. 0.194; difference = 0.033 [-0.023, 0.086]). This is likely because 89% of matched responses already contain such markers (Supplementary Section 6).

5.6 Selective Adjudication Simulation

The wide spread of code-level FP rates in the previous subsection suggests that selectively routing only disagreement-prone selections to researcher review could carry useful signal. The simulation in this subsection asks the workflow question directly: when the triage policy is fit on training responses and applied to held-out responses, how much review volume does it remove and how much retained participant work does it preserve?

Table 11: Retrospective selective-adjudication simulation on the matched subset, in-sample evaluation. The policy uses code-level FP-rate tertiles to route participant selections; the in-sample numbers should be read as a best-case bound and compared with the held-out numbers in Table 12. Neither evaluation is evidence of live-system performance.

Metric	Value	95% CI
Total selections	3207	
True positives (TP)	719	
False positives (FP)	2488	
Auto-accepted (low-FP tertile)	1725	
Flagged for review (mid + high)	1482	
Researcher time saved	0.538	[0.519, 0.558]
TP auto-accepted fraction	0.791	[0.760, 0.822]
FP auto-flagged fraction	0.535	[0.512, 0.557]
FP leaked (auto-accepted incorrectly)	0.465	[0.443, 0.488]
Flagging precision	0.899	[0.880, 0.915]
Flagging recall	0.535	[0.512, 0.557]
Flagging F1	0.671	[0.651, 0.689]

Policy: auto-accept low-FP tertile; flag mid & high for review.
53 codes with at least 3 selections; 338 matched responses.
Bootstrap 95% CIs: 2,000 resamples, seed = 42.

We applied a simple triage policy to participant selections on the matched subset. The policy uses code-level FP-rate groups:

- **Auto-accept**: selections on low-FP codes (bottom tertile). These are provisionally trusted without researcher review.
- **Flag for review**: selections on mid-FP codes (middle tertile).
- **Auto-flag**: selections on high-FP codes (top tertile). These are routed to researcher adjudication.

Codes with fewer than 3 selections in the relevant fitting set lack reliable FP-rate estimates and are conservatively routed to researcher review. The tertile thresholds come from the 53 codes with at least 3 selections, the same set used in the predictive model (Section 5.4). We report two evaluations: an in-sample evaluation, in which both tertile thresholds and triage outcomes are computed on the full matched subset, and a held-out evaluation, in which tertile thresholds are estimated from training responses only and applied to test responses (k=5 random response-fold split with seed = 42, plus a leave-one-participant-out variant; Supplementary Section 6.8).

Table 11 reports the in-sample evaluation as a best-case anchor. On the full matched subset, the policy auto-accepts 1,725 of 3,207 selections (53.8% [51.9, 55.8]), preserves 79.1% [76.0, 82.2] of selections researchers ultimately retained, and reaches a flagging precision of 0.899 [0.880, 0.915]. Table 12 reports the corresponding held-out numbers. Under k=5 response-fold cross-validation the policy still auto-accepts 53.7% [52.3, 54.9] of test selections, but the retained-selection preservation rate slips to 76.2% [74.7, 77.8] and flagging precision to 0.885 [0.876, 0.896]. Under leave-one-participant-out the auto-accept rate rises to 0.617 [0.590, 0.643] (because more codes fall under the conservative "not enough training data" route at the participant level), preservation drops to 0.663 [0.614, 0.710], and flagging precision drops to 0.737 [0.694, 0.782]. These differences quantify the

Table 12: Held-out evaluation of the triage policy on the matched subset under k=5 random response-fold cross-validation (seed = 42) and leave-one-participant-out (LOPO). Aggregate 95% CIs use 2,000 bootstrap resamples over per-fold metrics. The k=5 numbers are the headline workflow estimates; LOPO is a more conservative variant that tests transfer across participants.

Metric	In-sample	Held-out (k=5)	Held-out (LOPO)
Auto-accept rate	0.538	0.537 [0.523, 0.549]	0.617 [0.590, 0.643]
Retained-selection preservation	0.791	0.762 [0.747, 0.778]	0.663 [0.614, 0.710]
Flagging precision	0.899	0.885 [0.876, 0.896]	0.737 [0.694, 0.782]
Low-FP tertile retention (test)	0.330	0.317 [0.292, 0.343]	0.336 [0.303, 0.369]
Mid-FP tertile retention (test)	0.132	0.136 [0.118, 0.155]	0.115 [0.090, 0.140]
High-FP tertile retention (test)	0.033	0.077 [0.066, 0.090]	0.079 [0.051, 0.110]

k = 5 random response-fold split (seed = 42); LOPO leaves one participant out per fold.
Aggregate 95% CIs use 2,000 bootstrap resamples over the per-fold metric values (seed = 42).
Codes with <3 training selections are routed to review (no auto-accept).

gap between the in-sample best case and out-of-sample behavior; the held-out k=5 numbers are the headline workflow estimates we recommend reading.

As an independent check, we asked whether retained selections were more visible in participant explanations. We computed Jaccard overlap between each selection's open-text justification and the code's anchor text, meaning the written code description shown in the checklist. Retained selections showed higher textual overlap than rejected ones: mean Jaccard 0.051 vs. 0.028 (difference = 0.023 [0.017, 0.029]). This indicates that retained selections are more tightly grounded in what participants actually wrote.

This provides initial retrospective support for Design Hypothesis 4 (Section 6.4). Code-level divergence profiles carry enough information to motivate selective routing in this dataset, and the gap between the in-sample best case and the held-out estimates is itself informative for tool builders deciding how aggressively to trust auto-accept decisions. The simulation does not show that live adjudicators would change their decisions under this policy. A controlled adjudication study would provide the stronger test of whether routing changes researcher behavior.

5.7 Open-Text Evidence Analysis

The simulation suggests where review might be routed. It does not explain *why* abstract codes diverge more. The next analysis asks whether selections researchers did not retain still carry visible participant-authored evidence. If the gradient reflects different evidence standards, rejected selections on abstract codes with high disagreement should have less textual overlap with the code than rejected selections on concrete codes with low disagreement. We tested this by comparing word-overlap scores across the three FP-rate groups. The formal metric is Jaccard similarity: shared content words divided by all unique content words in the participant justification and code text, after removing common stop words.

The answer is consistent with the concreteness-gradient account but the cross-tertile difference is small. For concrete codes the evidence gap between retained and rejected selections is 0.022 [0.016, 0.029]. For mid-FP codes it is 0.022 [0.012, 0.033]. For the high-disagreement, more abstract tertile the point estimate is larger at 0.040 [0.005, 0.073], but the interval is wide (only 15 retained selections in this tertile) and the bootstrap difference between the high and low tertile gaps is +0.017 [-0.019, +0.052], which includes zero. We therefore read the per-tertile gaps as a directional pattern

Table 13: Open-text evidence by FP-rate tertile. The table reports mean Jaccard overlap for retained and rejected selections; higher overlap indicates closer wording, not proof of researcher agreement.

FP-rate tertile	Rejected (FP) n	Rejected (FP) Mean Jaccard	Retained (TP) n	Retained (TP) Mean Jaccard	Evidence gap [CI]
Low	1156	0.0283	569	0.0504	+0.0221 [0.0158, 0.0291]
Mid	890	0.0317	135	0.0533	+0.0216 [0.0117, 0.0325]
High	440	0.0213	15	0.0612	+0.0399 [0.0048, 0.0733]

Table 14: Rule-based textual-support check for sampled rejected selections. Twenty-three of 50 sampled cases met a stricter deterministic criterion; the sample comes from 113 high-disagreement rejected selections flagged by basic word overlap.

Metric	Value	Note
Cases sampled	50	Sampled from 113 high-FP overlap-screened cases
Rule-based textual support	23 (46%)	2+ overlap, or 1+ at/above TP 75th
Coincidental overlap	27 (54%)	Failed stricter overlap rule
Textual-support share	0.460	95% CI [0.320, 0.600]

rather than as a statistically distinguishable cross-tertile contrast. Among rejected selections, 40.7% of rejections on concrete codes with low disagreement have some textual overlap with the code, compared to 33.0% for rejections on abstract codes with high disagreement (Table 13).

This pattern is consistent with the epistemic gap hypothesis. Participants who select abstract codes may be coding from felt experience rather than from what they wrote. Concrete codes like "I prefer the vertical orientation" leave a textual trace that both participant and researcher can point to. Abstract codes like "Being able to zoom out and see all of the passages was useful" (43 selections, zero retained) describe a subjective impression that rarely appears verbatim in the open text.

Some rejected selections still contain useful textual evidence. Among the 440 rejected selections on high-FP codes, 113 (25.7% [20.2, 30.9]) had textual support at or above the median of retained selections. We call these *rejections with basic word overlap*: cases where the participant's open-text response may contain evidence that could support the code, even though researchers excluded the selection. Showing open-text alongside flagged codes during adjudication could therefore be worth testing for roughly one in four high-FP selections under this overlap-based screen.

5.8 Checking Flagged Rejections for Textual Support

The evidence analysis identifies 113 rejected selections on high-disagreement codes that showed basic word overlap. The remaining question is whether that overlap persists under a stricter rule or instead reflects accidental word matching. We checked this by sampling 50 of the 113 cases (seed = 42) and applying a stricter deterministic textual-overlap rule.

For each case, we computed whether the participant's open text contained enough content-word overlap with the code text. A case counted as having rule-based textual support if either condition held. First, it shared at least two content words with the code and reached the median Jaccard score for retained selections. Second, it shared at least one content word and reached the 75th percentile for retained selections. All remaining cases were classified as "coincidental overlap."

Table 14 reports the results. Of the 50 sampled cases, 23 (46.0% [32.0, 60.0]) contained enough matching content words to meet the stricter textual-support rule. The remaining 27 (54.0%) were classified as coincidental overlap under the rule.

This stricter check tempers the "roughly one in four" estimate from Section 5.7. Applying the sampled share to the full pool of high-FP rejected selections gives a product estimate of 0.118 (the screen pass rate of 0.257 [0.205, 0.310] times the stricter pass rate of 0.460 [0.320, 0.600]). Resampling matched responses for the screen factor and audit cases for the stricter factor (2,000 resamples, seed = 42) gives a joint bootstrap 95% CI of [0.078, 0.165] on the product, so the more honest summary is roughly 8–17% of rejected high-FP selections may carry rule-based textual support. Taken together, the retrospective simulation, overlap-based evidence screen, and sampled check of flagged rejected selections provide partial support for Design Hypothesis 4.

5.9 Interpretation

The 77.6% overall researcher-consensus FP rate tells a clear story. Participants and researchers use the same labels but apply different evidence standards. Participants select codes that match how the passage felt. Researchers select codes only when the written response provides specific textual support.

This evidence gap—not carelessness—is the most plausible account of the divergence. The evidence gap between retained and rejected selections is 0.022 [0.016, 0.029] for concrete codes and 0.040 [0.005, 0.073] for abstract codes; the bootstrap difference of +0.017 [-0.019, +0.052] is directional but spans zero, so we treat the cross-tertile contrast as suggestive rather than statistically robust. Rejected selections on abstract codes are also less likely to have any textual overlap with the code (33.0% vs. 40.7% for concrete codes). Abstract codes diverge more because participants know what they experienced but cannot always point to specific text. Concrete codes diverge less because both sides can point to the same observable features.

As secondary retrospective evidence, the exploratory random forest slightly outperforms the logistic baselines (leave-one-out cross-validated accuracy = 0.774, F1 = 0.818). Selection volume and unique selector count together account for the largest share of permutation importance in that model. This suggests that some divergence can be anticipated before adjudication begins, but only within this small retrospective sample, and we do not treat that model as the paper's primary evidence.

The practical consequence is that participatory coding tools should distinguish codes where participant self-coding is informative from codes where it needs researcher review. Concrete codes are informative. Abstract codes need review. Three analyses support this claim: the qualitative top-vs-bottom contrast, the selective-adjudication simulation (Section 5.6, with held-out preservation of 76.2% [74.7, 77.8] and held-out flagging precision of 0.885 [0.876, 0.896]), and the sampled check of flagged rejected selections (Section 5.8); the wide spread of code-level FP rates is the descriptive context that makes the routing problem worth solving. Together they suggest that divergence profiles can guide selective routing and that showing participant open-text may surface rule-based textual support in a subset of flagged cases.

6 Discussion

The analyses converge on one finding: participants and researchers code differently—not because participants are careless, but because they apply different evidence standards. Participatory tools need to work with this difference, not erase it. That gives tool designers three linked tasks: provisionally trust participant selections when historical divergence is low, route likely disagreements to review, and repair codebook language when participant wording exposes a gap.

6.1 Participant–Researcher Disagreement as a Distinct Signal

Prior work studies disagreement between trained researchers [11] or between researchers and AI [12]. Participant–researcher divergence is different. Participants apply codes based on what they experienced. Researchers apply codes based on what the text says. When two researchers disagree, they disagree about how to read the data. When a participant and a researcher disagree, one lived it and the other is reading about it.

The perspectivist literature [2, 5, 34] says this disagreement should not be treated uniformly as error. Our divergence far exceeds the moderate agreement (kappa = 0.46) Glazier et al. [15] found with a small category set. Codebook scale amplifies the gap.

Not all divergence has the same character. Divergence on abstract codes may be *irresolvable* [29]. Consider "Being able to zoom out and see all of the passages was useful" (43 selections, zero retained). The participant felt this was true. The researcher found no textual evidence. Deliberation may not resolve this gap when the participant's selection reflects lived experience that is not visible in the written response. Divergence on concrete codes is more likely *resolvable* through better definitions or examples.

This distinction has practical consequences. Tools that route all disagreement through the same pathway waste effort on resolvable cases and suppress valid perspectives on irresolvable ones. Reflexis [37] scaffolds deliberation for researcher teams but does not use participant signals. Our selective-adjudication simulation (Section 5.6) suggests that participant self-coding can serve as an additional input for such systems. The sampled check of flagged rejected selections (Section 5.8) adds specificity: among 50 sampled cases, 46.0% met the stricter textual-support rule. Surfacing this evidence during adjudication could help researchers tell resolvable disagreements from irresolvable ones.

6.2 Participant Preferences as Supporting Context

After using both input surfaces, 871 of 879 participants stated a preference (449 checklist, 422 open text). The near-even split masks different reasons. Checklist users describe the codebook as a recognition scaffold: "I had less ideas than what the options available had already given me" (Participant 71). Open-text users describe it as a constraint: "Choosing/entering the options filters the authenticity of responses" (Participant 54). These are complementary cognitive strategies: the checklist yields structured codes, open text yields nuanced justifications. Both support DH2 (preserve both surfaces). Full theme-by-preference results appear in Supplementary Section 5. The theme classification uses a keyword-based scheme (mean kappa = 0.887 between two deterministic classifiers), which shows scheme stability but not interpretive validity. We present these results as supporting context, not as a standalone finding (see Section 7).

6.3 Researcher Consensus as Complementary Discovery

The researcher workflow shows the same pattern at the coder level. Across 838 adjudicated responses, 77.3% of final positive response–code pairs came from exactly one coder before adjudication. Coder pairs rarely selected the same codes. Their median overlap (Jaccard index) was zero. The mean was 0.249 [0.234, 0.265] (Supplementary Table S25). This fits Pirolli and Card's [25] sense-making model: different analysts notice different things. Tools that show only what multiple coders already agree on will miss what each coder uniquely contributes.

6.4 What Should Tool Designers Do Differently?

We propose four design hypotheses grounded in the findings above. Of the four, Design Hypothesis 4 has the strongest evidence in this dataset: the selective-adjudication simulation (Section 5.6), the

open-text evidence analysis (Section 5.7), and the sampled check of flagged rejected selections (Section 5.8). The other three are motivated by the data but not experimentally tested.

DH1: Sequence open text before structured tagging. Showing open text before the checklist preserves unprimed language. Schwarz [31] showed that response alternatives frame survey answers. The 77.6% FP rate suggests that participants who see a structured codebook endorse codes that *resonate* with their experience rather than codes they can support with text. Open text first lets them state their reasoning before the codebook frames their choices.

DH2: Preserve both input surfaces. The near-even preference split shows that both surfaces serve real functions. Checklist users treat the codebook as a recognition scaffold. Open-text users treat it as a constraint on their voice (Section 6.2). Tools should let participants choose how much structure to add rather than requiring one input mode.

DH3: Design suggestion channels as codebook maintenance tools. Most suggestions are near-duplicates (46/87, 52.9%). The channel's value is exposing phrasing gaps, not generating novel themes. Each repair type points to a system response: synonym detection for reframing gaps, example-enriched definitions for specificity gaps, hierarchical expansion for granularity gaps.

DH4: Route adjudication selectively using divergence profiles. Code-level FP rates span a wide range across the 53 codes with at least 3 selections (mean FP rate 0.67 in the lowest tertile vs. 0.97 in the highest), and that spread is what makes triage worth running. The selective-adjudication simulation (Section 5.6) is the workflow test: under held-out k=5 response-fold cross-validation, a simple triage policy preserves 76.2% [74.7, 77.8] of selections researchers ultimately kept while reducing review volume by 53.7% [52.3, 54.9] (held-out flagging precision = 0.885 [0.876, 0.896]; in-sample numbers, 79.1% / 53.8% / 0.899, sit on the optimistic side of these intervals). The open-text evidence analysis (Section 5.7) adds a second layer: among rejected high-FP selections, 25.7% have textual support that could warrant reconsideration under the overlap-based screen. The sampled check (Section 5.8) estimates that 46.0% [32.0, 60.0] of those flagged cases meet the stricter textual-support rule.

The design recommendation is concrete but still prospective. First, compute code-level FP rates from an initial batch of dual-coded responses. Second, provisionally auto-accept low-FP codes and route high-FP codes to researcher review. Third, show participant open-text alongside flagged selections during adjudication. In held-out simulation this approach cuts review volume by 53.7% [52.3, 54.9] while preserving 76.2% [74.7, 77.8] of selections researchers ultimately kept. Under the current overlap-based screen, roughly 8–17% of rejected high-FP selections may contain rule-based textual evidence (joint bootstrap CI on the product, Section 5.8). That estimate should be treated as a candidate routing signal, not as evidence that live adjudicators would change their decisions.

This aligns with Schaekermann et al.'s [29] finding that deliberation improves consensus only for resolvable disagreements. Abstract codes warrant researcher attention. Concrete codes with low historical disagreement are the clearest candidates for provisional trust.

7 Limitations

This study analyzes qualitative response data from an educational passage-comparison task, not interview transcripts or fieldwork. The design hypotheses apply most directly to survey-like or reflection-based workflows. Whether the concreteness gradient generalizes to less structured paradigms (e.g., ethnographic coding, grounded theory) remains open.

The matched subset (338 responses) depends on researcher workflow progression, not random sampling. We cannot rule out coverage-dependent biases (Supplementary Section 8).

Researchers were blinded to participant self-codes during first-pass coding. The study therefore identifies where participant signals *could* support synthesis, not where they *do*. Testing causal claims about downstream analytic quality would require a controlled experiment.

The predictive models (Section 5.4) use only 53 codes. The exploratory random forest slightly outperforms the logistic baselines within this sample (leave-one-out cross-validated accuracy = 0.774, F1 = 0.818), but all predictive results remain constrained by only 53 code-level observations and the extended logistic model clearly overfits. A larger deployment would let us test individual features more reliably.

The preference coding (Section 6.2) uses a keyword-based scheme whose IRR (mean kappa = 0.887) compares two deterministic classifiers, not two human coders. A human-coded validation study would strengthen interpretive validity.

The corpus comes from a single-site educational deployment. Participants were students completing a course activity. Their coding behavior may partly reflect deference to researcher authority rather than purely different evidence standards. The qualitative concreteness contrast and the permutation-tested within-participant selection-rate variance (median CV = 0.166, reliably above the 0.143 produced by random reshuffling; Supplementary Section 6.9) argue against uniform random satisficing, but a low-stakes satisficing account cannot be fully ruled out.

The high-FP outcome used in the predictive model is defined by a 0.85 threshold on the per-code FP rate. Because codes with few selections can mathematically reach FP > 0.85 with much higher probability than codes with many selections, a portion of the dichotomized selection-volume effect is mechanically induced by the threshold rather than by participant behavior. The continuous-outcome variant in Section 5.4 and the threshold-sensitivity table in Supplementary Section 6.4 address this; the directional pattern survives both.

The selective-adjudication simulation (Section 5.6) reports both in-sample and held-out triage performance. The held-out k=5 numbers should be treated as the primary estimates; the in-sample numbers (53.8% / 79.1% / 0.899) are the optimistic best-case anchor and the leave-one-participant-out variant gives a more conservative bound (preservation 0.663 [0.614, 0.710], precision 0.737 [0.694, 0.782]). None of these is evidence of live-deployment performance. The sampled textual-support check (Section 5.8) uses an automated criterion rather than human expert judgment. The bootstrap 95% CI on the sampled textual-support share [32.0, 60.0] is wide. The 46.0% point estimate should be interpreted with caution. A larger sample would narrow the interval. A prospective study where adjudicators use the policy in real time—with and without open-text visible—would provide stronger evidence for DH4.

The same deployment, participants, and checklist apparatus are also analyzed in a sibling paper [1], which evaluates hybrid human–LLM alignment rather than participant codebook repair or participant–researcher divergence. The papers share collection context but make different analytic comparisons and contribution claims.

8 Positionality Statement

The research team designed the codebook, built the infrastructure, and ran adjudication. We are trained in qualitative methods within HCI and the learning sciences. This training shapes our perspective: we treat codebook design as an expert-authored activity and evaluate participant contributions against researcher-defined standards. Our labels—"near-duplicate," "meta-comment"—reflect a researcher-centric view. Participants might categorize their contributions differently. We mitigate this by analyzing participant reasoning in their own words (Section 6.2) and by reporting divergence symmetrically.

Participants were drawn from this multi-channel recruitment frame; not all participants were university students, and the qualitative-coding task was voluntary and uncompensated. The power

asymmetry in a recruitment context that combined an online crowd-work platform, intern cohorts in a multi-school online learning community, and an instructor channel—where participants may see the coding task as evaluative even when it is voluntary—should be considered when interpreting the preference justifications and suggestion behavior.

9 Ethics and Data Availability

The qualitative-coding deployment was reviewed by the authors' institutional review board and received a minimal-risk exempt determination (protocol number anonymized for review). Informed consent was obtained online before the first session via a written consent document; participation was voluntary and uncompensated, no sensitive or identifying information was collected, and participants could withdraw at any time without penalty. Sessions were recorded one-on-one online screen-share interviews; the screen-sharing requirement and the absence of compensation were disclosed in the consent document. The qualitative-coding session was embedded in a larger remote multi-school online learning community; admissions and instructor reviewers were blinded to condition assignments and task performance could not adversely affect admissions or course outcomes. Response data were de-identified before analysis. Participant identifiers in the dataset are pseudonymous codes, not real names. Participants consented to having de-identified information shared with other researchers; the derivative coding CSVs released with this submission contain no identifying information. The authors declare no conflicts of interest.

Use of generative AI.. Generative AI tools (OpenAI GPT-5.5 and Anthropic Claude Opus 4.7) assisted manuscript preparation, including language and LaTeX editing, analysis-script upkeep, consistency checking across artifacts, and packet verification automation. All AI-assisted outputs were reviewed and accepted by the human authors. AI tools did not collect participant data, did not produce final participant or researcher codes, and did not make final adjudication or interpretation decisions. The authors retain full responsibility for study design, analytic choices, interpretation of results, and the claims made in this paper.

Data availability. The anonymized supplementary reproducibility materials enumerated in Supplementary Section 9 list the de-identified derivative datasets and analysis scripts needed to regenerate the manuscript artifacts. Consistent with the institutional data-governance constraints described in that section, these materials are not attached to the initial submission packet; they are available to reviewers upon editor request during the review cycle and will be released through the venue's supplementary-materials channel after acceptance. Running `make figures` is the authoritative route for regenerating all manuscript-facing tables and figures from those materials once they are released. The shared materials are sufficient for manuscript-facing regeneration, subject to the review-only data-governance and third-party asset restrictions stated here. They are not a release of the institution-governed source records from the live educational deployment. Access to the underlying source records remains governed by the educational data governance policies of the authors' institution. The derivative CSVs use pseudonymous identifiers, retain open text only where needed for manuscript reproduction, and are accompanied by field manifests and direct-identifier scan summaries in the supplementary reproducibility materials. The Brysbaert et al. [4] concreteness norms used by our analysis are not redistributed with these materials; they are available from the publisher's electronic supplementary material accompanying that paper and from the authors' canonical hosting page.

10 Conclusion

When participants code their own qualitative data, they disagree with researchers—but the disagreement is structured and predictable. This paper's central finding is the *concreteness gradient*: abstract,

experiential codes produce high participant–researcher divergence, while concrete, text-grounded codes produce lower divergence. Code-level FP rates span a wide range across the 53 codes with at least 3 selections (mean 0.67 in the lowest tertile vs. 0.97 in the highest), and a separate qualitative inspection of the most- and least-divergent codes shows that the highest-divergence codes describe felt experience while the lowest-divergence codes describe more text-grounded features.

An open-text evidence analysis is consistent with the mechanism behind this gradient. The evidence gap between retained and rejected selections is 0.022 [0.016, 0.029] for concrete codes and 0.040 [0.005, 0.073] for abstract codes; the bootstrap difference is +0.017 [-0.019, +0.052], directional but not statistically distinguishable from zero, so we treat the cross-tertile contrast as suggestive rather than confirmed. A sampled check of 50 flagged rejected selections finds that 23 meet stricter deterministic textual-support criteria for the code. A retrospective selective-adjudication simulation shows that routing by divergence profile preserves 76.2% [74.7, 77.8] of selections researchers ultimately kept while reducing reviewed selections by 53.7% [52.3, 54.9] under held-out k=5 response-fold cross-validation (in-sample 79.1% / 53.8%). A small retrospective model comparison was directionally consistent with the same ordering, but it serves only as auxiliary support within a 53-code sample.

A supporting finding is that participant code suggestions work primarily as targeted codebook repair. Of 87 suggestions, 46 (52.9%) were near-duplicates that exposed phrasing gaps, not missing themes. Participants split nearly evenly between checklist and open text, but for different reasons: checklist users value memory retrieval, open-text users value expressiveness.

These findings ground four design hypotheses for participatory coding tools: sequence open text before structured tagging, preserve both input surfaces, design suggestion channels as codebook maintenance tools, and route adjudication selectively based on divergence profiles. Together, the hypotheses organize the design problem around trust, routing, and repair. Design Hypothesis 4 receives initial retrospective support from the triage simulation (held-out k=5 flagging precision = 0.885 [0.876, 0.896]; in-sample 0.899), the open-text evidence analysis, and the sampled rule-based check of flagged rejected selections.

For researchers building participatory coding systems, the actionable takeaway is prospective rather than deployment-validated. Treat code-level divergence profiles as a candidate routing signal for separating codes that need human review from codes that may be provisionally trusted from the participant signal. In held-out simulation, this selective approach reduces review volume by 53.7% [52.3, 54.9] while preserving 76.2% [74.7, 77.8] of selections researchers ultimately kept (in-sample 53.8% and 79.1% are the optimistic best-case anchors). When open-text is available, surface it alongside flagged selections. The sampled textual-support check, propagated through a joint bootstrap CI on the product of the screen-pass and stricter-rule rates, suggests that roughly 8–17% of rejected high-FP selections carry rule-based textual support that could warrant adjudicator attention.

The broader lesson for CSCW is this: participant–researcher disagreement is not noise. It follows a predictable structure that tools can test as a workflow signal. Concrete codes with low historical disagreement appear more amenable to provisional trust in retrospective analysis. Abstract codes with high historical disagreement appear to need researcher adjudication. Tools that recognize this structure—differentiating by divergence profile, surfacing open-text evidence, and routing review selectively—can integrate participant voice without collapsing the evidence gap that makes it distinctive.

References

[1] Anonymous. 2026. The Epistemic Triad: Auditing Participant-Endorsed Selections in Hybrid Human–LLM Coding. Under review at a peer-reviewed venue; submitted as supplementary material.

[2] Valerio Basile, Michael Fell, Tommaso Fornaciari, Dirk Hovy, Silviu Paun, Barbara Plank, Massimo Poesio, and Alexandra Uma. 2021. We Need to Consider Disagreement in Evaluation. In *Proceedings of the 1st Workshop on Benchmarking: Past, Present and Future (BPPF 2021)*. Association for Computational Linguistics, Online, 15–21. doi:10.18653/v1/2021.bppf-1.3

[3] Linda Birt, Suzanne Scott, Debbie Cavers, Christine Campbell, and Fiona Walter. 2016. Member Checking. *Qualitative Health Research* 26, 13 (2016), 1802–1811. doi:10.1177/1049732316654870

[4] Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness Ratings for 40 Thousand Generally Known English Word Lemmas. *Behavior Research Methods* 46, 3 (2014), 904–911. doi:10.3758/s13428-013-0403-5

[5] Federico Cabitza, Andrea Campagner, and Valerio Basile. 2023. Toward a Perspectivist Turn in Ground Truthing for Predictive Computing. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence*. AAAI Press, Washington, DC, 6860–6868. doi:10.1609/aaai.v37i6.25840

[6] Natalie A. Carlson and Vanessa Burbano. 2026. The use of LLMs to annotate data in management research: Foundational guidelines and warnings. *Strategic Management Journal* 47, 3 (2026), 699–725. doi:10.1002/smj.70023

[7] Bonnie Chinh, Ge Gao, Hana Habib, Leah Ajmani, and Cecilia R. Aragon. 2019. Ways of Qualitative Coding: A Case Study of Four Strategies for Resolving Disagreements. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, Glasgow, Scotland, UK, 1–6. doi:10.1145/3290607.3312879

[8] Melanie Feinberg. 2017. A Design Perspective on Data. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, Denver, CO, USA, 2952–2963. doi:10.1145/3025453.3025837

[9] Daniel Filonik, Tomasz Bednarz, Markus Rittenbruch, and Marcus Foth. 2015. Collaborative Data Exploration Interfaces - From Participatory Sensing to Participatory Sensemaking. In *2015 Big Data Visual Analytics (BDVA)*. IEEE, Hobart, TAS, Australia, 1–2. doi:10.1109/BDVA.2015.7314289

[10] Miranda Fricker. 2007. *Epistemic Injustice: Power and the Ethics of Knowing*. Oxford University Press, Oxford, UK. doi:10.1093/acprof:oso/9780198237907.001.0001

[11] 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, Article 132 (2018), 24 pages. doi:10.1145/3274401

[12] Jie Gao, Kenny Tsu Wei Choo, Junming Cao, Roy Ka-Wei Lee, and Simon Perrault. 2023. CoAIcoder: Examining the Effectiveness of AI-Assisted Human-to-Human Collaboration in Qualitative Analysis. *ACM Transactions on Computer-Human Interaction* 31, 1 (2023), 1–38. doi:10.1145/3617362

[13] Jie Gao, Yuchen Guo, Gionnieve Lim, Tianqin Zhang, Zheng Zhang, Toby Jia-Jun Li, and Simon Tangi Perrault. 2024. CollabCoder: A Lower-Barrier, Rigorous Workflow for Inductive Collaborative Qualitative Analysis with Large Language Models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu, HI, USA, 1–29. doi:10.1145/3613904.3642002

[14] Simret Araya Gebreegziabher, Zheng Zhang, Xiaohang Tang, Yihao Meng, Elena L. Glassman, and Toby Jia-Jun Li. 2023. PaTAT: Human-AI Collaborative Qualitative Coding with Explainable Interactive Rule Synthesis. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg, Germany, 19 pages. doi:10.1145/3544548.3581352

[15] Rebecca A. Glazier, Amber E. Boydstun, and Jessica T. Feezell. 2021. Self-coding: A method to assess semantic validity and bias when coding open-ended responses. *Research & Politics* 8, 3 (2021), 1–8. doi:10.1177/20531680211031752

[16] Hanne De Jaegher and Ezequiel Di Paolo. 2007. Participatory sense-making: An enactive approach to social cognition. *Phenomenology and the Cognitive Sciences* 6, 4 (2007), 485–507. doi:10.1007/s11097-007-9076-9

[17] Daye Kang, Zhuolun Han, Jiahe Tian, Muhan Zhang, and Jeffrey M. Rzeszotarski. 2025. ThemeViz: Understanding the Effect of Human-AI Collaboration in Theme Development with an LLM-Enhanced Interactive Visual System. *Proceedings of the ACM on Human-Computer Interaction* 9, CSCW, Article 494 (2025), 29 pages. doi:10.1145/3757675

[18] Malik Khadar, Daniel Runningen, Julia Tang, Stevie Chancellor, and Harmanpreet Kaur. 2025. Wisdom of the Crowd, Without the Crowd: A Socratic LLM for Asynchronous Deliberation on Perspectivist Data. *Proceedings of the ACM on Human-Computer Interaction* 9, 7, Article CSCW526 (2025), 35 pages. doi:10.1145/3757707

[19] Jon A. Krosnick. 1999. Survey Research. *Annual Review of Psychology* 50 (1999), 537–567. doi:10.1146/annurev.psych.50.1.537

[20] Kehua Lei, Aidan Ladenburg, Zahra Kais Petiwala, Zili Wang, Dishita Jhawar, Ipsita Bisht, Ansh Kumar, and David T. Lee. 2025. Dynamic Surveys: Using LLMs to Blend Qualitative Depth, Quantitative Structure, and Collaborative Interaction. *Proceedings of the ACM on Human-Computer Interaction* 9, 7, Article CSCW405 (2025), 26 pages. doi:10.1145/3757586

[21] Yvonna S. Lincoln and Egon G. Guba. 1985. *Naturalistic Inquiry*. Sage Publications, Newbury Park, CA. https://collegepublishing.sagepub.com/products/naturalistic-inquiry-1-842

[22] Michael Muller, N. Sadat Shami, Stephanie Boden Stubbe, Werner Geyer, and Jonathan Grudin. 2012. Return on Contribution (ROC): A Metric for Enterprise Social Software. In *Proceedings of the 2012 European Conference on Computer Supported Cooperative Work*. Springer, London, UK, 1–20. doi:10.1007/978-1-4471-4093-1_1

[23] Melanie Nind. 2011. Participatory data analysis: a step too far? *Qualitative Research* 11, 4 (2011), 349–363. doi:10.1177/1468794111404310

[24] Cassandra Overney, Belén Saldías, Dimitra Dimitrakopoulou, and Deb Roy. 2024. SenseMate: An Accessible and Beginner-Friendly Human-AI Platform for Qualitative Data Analysis. In *Proceedings of the 29th International Conference on Intelligent User Interfaces (IUI '24)*. ACM, New York, NY, 922–939. doi:10.1145/3640543.3645194

[25] Peter Pirolli and Stuart Card. 2005. The Sensemaking Process and Leverage Points for Analyst Technology as Identified through Cognitive Task Analysis. *Proceedings of the International Conference on Intelligence Analysis* 5 (2005), 2–4.

[26] Barbara Plank. 2022. The "Problem" of Human Label Variation: On Ground Truth in Data, Modeling and Evaluation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 10671–10682. doi:10.18653/v1/2022.emnlp-main.731

[27] Andrew Prahl. 2026. The AI-Reflexivity Checklist (ARC): A Pre-Analysis Pause for LLM-Assisted Coding. *Qualitative Health Research* 36, 2-3 (2026), 181–190. doi:10.1177/10497323251401503

[28] Daniel M. Russell, Mark J. Stefik, Peter Pirolli, and Stuart K. Card. 1993. The cost structure of sensemaking. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*. ACM, New York, NY, 269–276. doi:10.1145/169059.169209

[29] 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, Article 154 (2018), 19 pages. doi:10.1145/3274423

[30] Hope Schroeder, Madeleine Aubin Le Quéré, Cass Randazzo, David Mimno, and Sarita Schoenebeck. 2025. Large Language Models in Qualitative Research: Uses, Tensions, and Intentions. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, Yokohama, Japan, 1–17. doi:10.1145/3706598.3713120

[31] Norbert Schwarz. 1999. Self-Reports: How the Questions Shape the Answers. *American Psychologist* 54, 2 (1999), 93–105. doi:10.1037/0003-066X.54.2.93

[32] Ansh Sharma, Karen Cochrane, and James R. Wallace. 2025. DeTAILS: Deep Thematic Analysis with Iterative LLM Support. arXiv preprint arXiv:2510.17575. doi:10.48550/arXiv.2510.17575

[33] Alexandra N. Uma, Tommaso Fornaciari, Dirk Hovy, Silviu Paun, Barbara Plank, and Massimo Poesio. 2021. Learning from Disagreement: A Survey. *Journal of Artificial Intelligence Research* 72 (2021), 1385–1470. doi:10.1613/jair.1.12752

[34] Ruyuan Wan, Haonan Wang, Ting-Hao Kenneth Huang, and Jie Gao. 2025. From Noise to Nuance: Enriching Subjective Data Annotation through Qualitative Analysis. In *Proceedings of the Fourth Workshop on Bridging Human–Computer Interaction and Natural Language Processing (HCI+NLP)*. Association for Computational Linguistics, Suzhou, China, 240–254.

[35] Meredith Woods, Trena Paulus, David P. Atkins, and Rob Macklin. 2016. Advancing Qualitative Research Using Qualitative Data Analysis Software (QDAS)? Reviewing Potential Versus Practice in Published Studies using ATLAS.ti and NVivo, 1994–2013. *Social Science Computer Review* 34, 5 (2016), 597–617. doi:10.1177/0894439315596311

[36] Yinuo Xu and David Jurgens. 2026. Beyond Consensus: Perspectivist Modeling and Evaluation of Annotator Disagreement in NLP. arXiv preprint arXiv:2601.09065. doi:10.48550/arXiv.2601.09065

[37] Runlong Ye, Oliver Huang, Patrick Yung Kang Lee, Michael Liut, Carolina Nobre, and Ha-Kyung Kong. 2026. Reflexis: Supporting Reflexivity and Rigor in Collaborative Qualitative Analysis through Design for Deliberation. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems*. ACM, Barcelona, Spain, 18 pages. doi:10.1145/3772318.3791275