Impact of Human-AI Interaction on User Trust and Reliance in AI-Assisted Qualitative Coding

JIE GAO, Singapore University of Technology and Design, Singapore

JUNMING CAO, Fudan University, China

SHUNYI YEO, KENNY TSU WEI CHOO, Singapore University of Technology and Design, Singapore

ZHENG ZHANG, TOBY JIA-JUN LI, University of Notre Dame, USA

SHENGDONG ZHAO, National University of Singapore, Singapore

SIMON TANGI PERRAULT, Singapore University of Technology and Design, Singapore

While AI shows promise for enhancing the efficiency of qualitative analysis, the unique human-AI interaction resulting from varied coding strategies makes it challenging to develop a trustworthy AI-assisted qualitative coding system (*AIQCs*) that supports coding tasks effectively. We bridge this gap by exploring the impact of varying *coding strategies* on *user trust and reliance on AI*. We conducted a mixed-methods split-plot 3 × 3 study, involving 30 participants, and a follow-up study with 6 participants, exploring varying *text selection* and *code length* in the use of our *AIQCs* system for qualitative analysis. Our results indicate that qualitative open coding should be conceptualized as a series of distinct subtasks, each with differing levels of complexity, and therefore, should be given tailored design considerations. We further observed a discrepancy between perceived and behavioral measures, and emphasized the potential challenges of under- and over-reliance on *AIQCs* systems. Additional design implications were also proposed for consideration.

CCS Concepts: • **Human-centered computing** → **Empirical studies in collaborative and social computing**.

Additional Key Words and Phrases: Human-AI Interaction, Qualitative Analysis, Qualitative Coding Strategies, Trustworthiness, Reliance, Helpfulness

1 INTRODUCTION

Qualitative coding is a fundamental tool within qualitative research [14, 17, 54]. It is the initial procedure that converts raw data into a format for subsequent stages of analysis. Despite its importance, coding remains a laborious and frequently repetitive process requiring multiple iterations. In response, researchers have sought to alleviate this labor-intensive task by harnessing Artificial Intelligence (AI), leading to the development of the *AI-assisted Qualitative Coding system (*AIQCs*)* [15, 25, 27, 37, 51]. Primarily, AI can facilitate the coding process by offering suggestions informed by past coding annotations, thus prompting users to consider alternative perspectives and rephrase their codes, particularly during the early coding stages.

On the other hand, trust is fundamental in constructing human-centric AI systems [5, 58]. Jiang et al. [28] have underscored the multitude of factors that foster distrust between humans and AI within the context of qualitative analysis. These include skepticism towards the AI's capability to execute qualitative analysis reliably, noticeable behavioral disparities between humans and AI, the absence of explanations of AI suggestions, and so on.

However, we argue that varying human-AI interactions in qualitative analysis, which arise from different coding strategies—a factor that has seemingly been overlooked—pose unique challenges for AI to consistently provide reliable suggestions throughout the *Open Coding* process. Depending on the final objective, coding approaches can vary greatly. For instance, some might opt to code entire paragraphs with a concise code, providing a broad classification for large text segments. Conversely, others might focus on phrases, applying lengthier codes for more detailed insights. Despite similar processes, these strategies yield vastly different depths and scopes in coding outcomes.

In particular, there is a cascading chain of influence—from the interaction between humans and AI to the user trust and reliance: the ① **human-AI interaction** can substantially vary based on the ② **coding strategies** employed. It influences the ③ **users' input**—essentially, the ④ **training data for the AI**—which consequently impacts both the ⑤ **model's performance** and, therefore, the ⑥ **quality of AI suggestions**. This chain of influences, in turn, shapes ⑦ **humans' perceived trust and reliance** in AI systems. Therefore, our objective is to bridge this gap by examining this influence chain within the context of qualitative analysis.

Trust is a concept with many definitions [58]. For this work, we specifically focus on users' behavioral trust (or reliance), perceived trustworthiness and helpfulness of *AIQCs*. We selected these facets because current trust-related research in *AIQCs* primarily addresses users' perceived trustworthiness [28]. Moreover, discrepancies between perceived and behavioral trust are common in AI-assisted tasks [20, 44, 58, 59], highlighting the need for a holistic understanding of trust. Additionally, it's pertinent to investigate whether imperfect AI can still offer valuable helpfulness, considering the innate complexity of achieving perfection in subjective tasks such as qualitative analysis [15, 16, 30].

We then operationalize different coding strategies by controlling the coding granularity and introduce two factors: *Text Granularity* and *Code Granularity*. We specifically aim to understand how varying coding granularity influences the following aspects:

RQ1. How does coding granularity impact the model performance of *AIQCs*?

RQ2. How does coding granularity impact users' *Decision Time* and *Coding Behavior* when using *AIQCs*?

RQ3. How does coding granularity impact users' *Behavioral Trust* (i.e., reliance)?

RQ4. How does coding granularity impact users' *Perceived Trustworthiness* and *Helpfulness* of *AIQCs*?

RQ5. How does coding granularity impact users' *Subjective Preferences* when using *AIQCs*?

In response to these research questions, we carried out a split-plot study involving 30 participants, supplemented by a follow-up study with 6 participants. During the main study, participants were given the task of coding texts with varying granularities utilizing individual AI models. We also collected supplemental data wherein users performed the same tasks without access to the AI models for comparative analysis.

Our findings suggested that qualitative coding should not be perceived as a uniform task, but as a collection of subtasks with varying levels of difficulty. Certain subtasks were found to be more challenging (*Paragraph*, *Long Codes*), whereas others were comparatively simpler (*Short Codes*, *Mixed Codes*, *Sentence*, *Selective*). An intriguing discrepancy between perceived and behavioral measures emerged from our study: participants indicated higher *Perceived Helpfulness* for more difficult tasks as compared to simpler ones, but exhibited lower *Behavioral Trust*; on the contrary, for simpler tasks, participants demonstrated higher *Behavioral Trust* but lower *Perceived Helpfulness*. Our study additionally highlighted the potential pitfalls of both under-reliance and over-reliance on *AIQCs*. Under-reliance might hinder users from fully exploiting the benefits of *AIQCs*, while over-reliance could lead to ostensibly focused yet shallow outcomes. These factors necessitate careful deliberation in the design of trustworthy *AIQCs*.

The contribution of this work is two-fold:

- The results of a user study that sheds light on how human-AI interaction in qualitative analysis impacts *AIQCs*'s model performance, user trust, reliance, and perceived helpfulness, with the varying difficulty of *Open Coding* subtasks.
- A series of design principles focused on key factors to consider when designing *AIQCs* to ensure appropriate reliance, trustworthiness and helpfulness.

2 BACKGROUND

Qualitative coding is a key tool to analyze qualitative data. Charmaz [14] presented two phases in the Grounded Theory: *Initial Coding* (or *Open Coding*) and *Focused Coding*. Serving as the preliminary step in transitioning from raw data's concrete ideas and concepts to formulating analytic interpretations [14], *Open Coding* involves assigning a summarizing label to varied segments of data, with sizes ranging from a single word to a full paragraph [14, 17, 19, 52]. Subsequently, these labels or codes undergo thorough discussion within a team, leading to the development of a codebook. This codebook comprises a variety of labels/codes correlating with the raw data, thereby facilitating further data analysis [19].
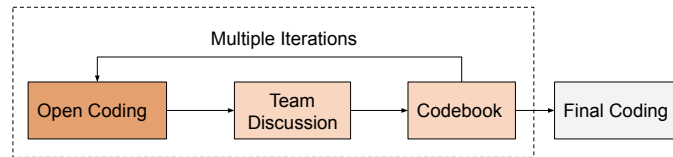

Fig. 1. A Circular Coding Process (See More in [19, 50, 52]).

Nonetheless, the coding process is labor-intensive and demands significant effort, often necessitating multiple iterations within a team [37, 51, 62]. This is because the development of the codebook is not a linear process but rather cyclical in nature (see Figure 1).

In particular, *Open Coding* acts as a key step in this development process [14, 19] and is inevitably revisited multiple times. Often, researchers need to revisit raw data, potentially collecting more data and conducting additional coding until reaching saturation, ensuring no nuanced information is overlooked.

Therefore, the demanding *Open Coding* process, one of the major causes of slow coding progress, has spurred the development of *AIQCs*. Our research focuses on this process, exploring how various factors influence AI's ability to provide valuable helpfulness for *Open Coding*.

3 RELATED WORK

3.1 AI-assisted Qualitative Coding Systems

Many recent studies have explored the application of (semi)automated techniques to facilitate qualitative analysis. Some of these studies [18, 37, 46] suggest utilizing code rules for extracting pertinent sections from a given text. For instance, a Boolean rule for `Definition of arts` may be constructed by linking various keywords using Boolean operators such as AND, OR, and NOT (e.g., `(definition OR define OR constitute) AND art`) [37]. This rule is then evaluated against a target text, and a match is identified if their similarity surpasses a specified threshold.

Moreover, scholars propose code pattern auto-detection in order to support more flexibility [25, 41]. For instance, Nelson's three-step method [41] applies unsupervised machine learning for data pattern discovery, enhancing scalable, exploratory analysis. Meanwhile, PaTAT, introduced by Gebreegziabher et al. [25], finds user

coding patterns in real-time, predicting future codes. These studies indicate that auto-detection of code rules for partial automation shows significant potential.

Furthermore, unsupervised machine learning approaches, such as topic modeling [7, 23, 27, 33], have proven valuable for detecting topics or labels in qualitative data, especially when dealing with large-scale datasets. By identifying statistical regularities within text, topic modeling can discern thematic patterns, yielding results akin to traditional grounded theory methods [33, 40]. This approach allows researchers to uncover topics or labels during the early stages of qualitative analysis more effectively [23, 29, 42].

In addition, supervised techniques such as text classification has gained widespread usage in qualitative analysis. For instance, Yan et al. [62] utilized Support Vector Machine (SVM) classification, using pre-selected features and parameters. They trained the SVM model with codes provided by human coders to classify large-scale text data. In a similar vein, Rietz et al.'s Cody [51] used a logistic regression model, employing stochastic gradient descent (SGD) learning. This model was trained to categorize unseen data based on existing annotations.

While substantial research exists in this field, a comprehensive examination of user reliance, trustworthiness and helpfulness of AI-assisted systems remains scarce [28]. To our knowledge, this work represents the first attempt to deeply explore how different human-AI interaction strategies within qualitative analysis impact the user trust and reliance for *AIQCs*.

3.2   Trust with *AIQCs*

Trust encompasses various definitions [20, 32, 38, 58]. We concentrate on three closely related concepts of trust: *Perceived Trustworthiness*, *Behavioral Trust* (i.e., reliance), and *Perceived Helpfulness*. Firstly, *Perceived Trustworthiness* is the perception of a system's trustworthiness, defined as "the extent to which the trustee believes that an automated system will behave as expected" [44, 58]. We focused on *Perceived Trustworthiness*, because according to a study by Jiang et al. [28], it was observed that participants generally perceived the trustworthiness of AI in qualitative analysis to be very low. Secondly, *Behavioral Trust* refers to the act of following or accepting someone's recommendation [20, 44, 58, 59] and is often interchangeable with reliance [20, 32, 58]. Our tasks fall under the category of AI-assisted decision making, where AI supports users in reaching their final decisions. As users are the ultimate decision makers, researchers usually measure users' reliance and behavior objectively. Conversely, we chose to study both *Behavioral Trust* and *Perceived Trustworthiness* because researchers also identified a discrepancy between users' perceived and behavioral trust [13, 44, 53, 58]. By observing both subjective and objective aspects, we can gain a more comprehensive understanding of users' overall trust in the system [9, 13, 53]. Lastly, *Perceived Helpfulness* is described as "the extent to which users perceive the recommendation as being capable of facilitating judgment or decisions" [34, 48]. There is a strong connection between *Perceived Trustworthiness* and *Perceived Helpfulness*: if users find recommendations helpful, they are more likely to seek advice from those recommendations, thereby fostering trust in the system's capabilities. In addition, we aim to investigate the potential benefits of less trustworthy AI and imperfect system [30] in subjective tasks [15, 16] and the pitfalls of seemingly perfect and trustworthy systems [6].

As trust has become a significant topic in the fields of CSCW and HCI [5, 6, 58], there is also a growing interest in exploring and establishing this element within the *AIQCs* domain. In the interview conducted by Jiang et al. [28], the authors highlight several sources that contribute to distrust in AI. For instance, they point out discrepancies in the "typical behavior of humans and AI", as AI offers direct suggestions even when humans cannot provide a specific "correct" recommendation, and AI tends to prioritize suggestions with higher probabilities rather than subtle and nuanced insights. They also pointed out that low-precision models often require extra human effort for corrections. Moreover, the absence of explanations from AI and skepticism regarding AI's capacity for creativity and serendipity frequently lead to increased distrust. On the contrary, excessive reliance on AI might prompt researchers to defer to AI as the ultimate authority, potentially compromising human deliberation

in the decision-making process. Building on the work of Jiang et al., we delve deeper into trust issues between humans and AI within this domain. We anticipate that in *AIQCs*, users should establish an appropriate level of trust, avoiding both under- and over-reliance.

While current *AIQCs* approaches typically rely on human input for model training and generating code suggestions, there is a lack of research examining the human-AI interactions within the context of *AIQCs*. These interactions may introduce unique challenges specific to *AIQCs* that can significantly influence user trust and reliance. Certain interactions have the potential to generate higher-quality input, leading to more accurate code suggestions and improved assistance, while others may have a detrimental effect on these outcomes, subsequently shaping user perceptions and ultimate reliance on the systems.

3.3 Human-AI interaction within AIQCs

There has been significant interest in finding ways for end-users, rather than experts, to interact meaningfully with machine learning systems in order to enhance system performance and user experience [2, 55]. Researchers have also investigated how to refine features of machine learning systems by incorporating human perspectives, particularly in complex qualitative content analysis scenarios [35]. For *AIQCs*, AI often relies on human-generated training data, which serves as model input. However, unlike many traditional AI tasks, the inherently subjective nature of qualitative analysis poses a unique challenge. This challenge stems from the difficulty of obtaining specific and consistent human inputs, such as labels and text, for AI models. A major contributing factor to this issue is the variability in code granularity present in human coding [36, 52]. This variability may lead to inconsistencies and unreliability in the produced data, subsequently affecting the performance and trustworthiness of *AIQCs*.

Text Granularity denotes the specific selections of text that are to be coded. Imagine an *Open Coding* exercise where coding occurs on a word-by-word basis. In such a scenario, the overall context of the text is lacking, making it difficult for the model to suggest any useful codes. Consequently, coders may lose trust in the AI's ability and decide to stop using the system. On the other hand, performing line-by-line or sentence-by-sentence coding would provide the AI with more context. This increased context could potentially enhance the performance of AI models [22], thereby influencing the system's perceived trustworthiness among users. In this work, we have chosen to examine three different levels of text granularity: **sentence, paragraph, and selective**. For the last level, users can select phrases of any length, which more closely resembles a regular *Open Coding* process.

Code Granularity refers to the length and specificity of a code [36]. When a code is short, broad, and general (e.g., "experience", "leadership"), the AI might exhibit commendable performance from a classification perspective: the probability of AI suggestions aligning with the user-selected text is elevated, thereby expanding the pool of potential choices within the AI's suggestions. Despite this, dependency on AI assistance under these circumstances is not advisable. There's a risk of users becoming overly dependent on it, which may undermine the depth and diversity of qualitative analysis. On the other side, if users add excessively lengthy or detailed codes (e.g., "he hosts lots of activities", "her pets are very cute"), they may not serve as suitable categories for classification as they could exhibit limited commonality for code reuse. This situation could potentially impact the performance of the AI models. In this work, we have elected to explore three distinct levels of code granularity: **Short Codes** (i.e., Concise and General Codes), **Long Codes** (i.e., Detailed and Comprehensive Codes), and **Mixed Codes** (Natural Codes). In the case of the latter, users may employ code lengths ranging from one to six words, more closely mirroring a typical *Open Coding* process.

Both of the aforementioned scenarios could hinder the AI model from performing as well as expected. As a result, AI could become less useful, leading users to either under-utilize it or rely on it excessively. Hence, the granularity levels for both codes and text selections must be carefully designed, to enhance the clarity and

Fig. 2. *AIcoder* Interface. The above figure shows a user was doing coding using *Mixed Codes*. The user can add codes by 1) selecting the text of significance or interest, including phrases, sentences or paragraphs, etc.; 2) clicking the comment button to create a code; 3) typing new code or selecting code suggestions suggested by AI. Each code also shows the confidence level, ranging between 0 and 1; 4) code is shown beside the selected text; 5) edit codes.

consistency of the coding scheme [16, 37, 51]. We anticipate that our evaluation will distill key insights, thus aiding in the creation of trustworthy, reliable, and beneficial *AIQCs*.

4   AICODER

We developed a prototype, *AIcoder*, which adopts an approach similar to prior research that treats qualitative coding as a classification task [51, 62]. The user interface of *AIcoder* is displayed in Figure 2, while the backend structure is depicted in Figure 3. With *AIcoder*, users can conveniently highlight any segment of text and assign a specific code. Following the coding, the selected text snippets and their corresponding codes are utilized to fine-tune a model based on their inputs. Ultimately, the user can highlight another piece of text to receive several recommendations from the model. We outline the components of *AIcoder* in the following sections.

4.1   Interface

The interface (see Figure 2) is built on 1) *Etherpad*[1], an open source web text editor, supporting users editing text online [3, 8, 26], and 2) its plugin, *ep_comment_pages*[2], supporting adding comments beside the text. With the interface, users can add codes, review code history, modify previous entries, and discard unnecessary codes. In

---

[1]https://etherpad.org/
[2]https://github.com/ether/ep_comments_page

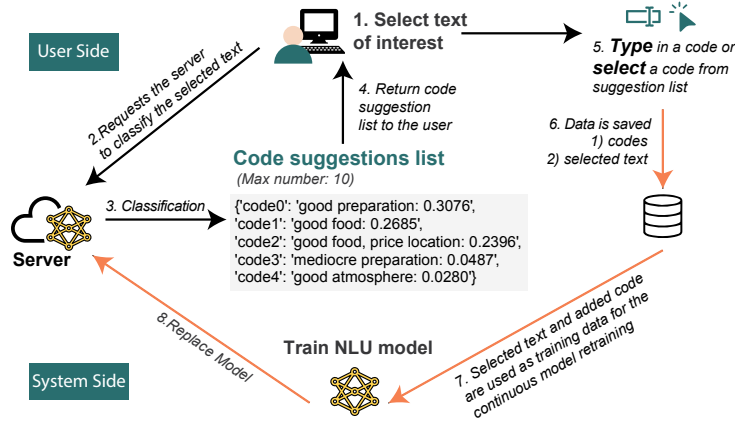Fig. 3. Process of recommendation generation. User side: 1) the user selects the text and clicks on "comment" button; the system 2) automatically requests suggestions from the model server, 3) conducts a classification process, 4) returns a list containing up to 10 code suggestions for the user to either select from or refer to, and 5) the user decides to either create their own codes or select one from the list. System Side: 6) the codes and labeled text are subsequently stored for future use, 7) the selected text and added codes are reused as training data to fine-tune a new model, and 8) the updated model is subsequently deployed onto the server.

addition, *ep_comment_pages* has been customized to provide a list of code suggestions ($n \leq 10$) upon user request. These suggestions are ranked based on their confidence level, which falls within the range of 0 to 1, indicating the cosine similarity score between the predicted labels and the corresponding text [3].

4.2 AI Model

The AI model utilized in *AIcoder* is based on the NLU component of Rasa[4], an open-source Python machine learning framework. Specifically, we utilize a Rasa-recommended NLU pipeline to train an NLU classification model[5]. This includes several components: SpacyNLP, SpacyTokenizer, SpacyFeaturizer, RegexFeaturizer, LexicalSyntacticFeaturizer, two instances of CountVectorsFeaturizer, and DIETClassifier. Within this pipeline, we selected the pre-trained SpacyNLP language model "en_core_web_trf"[6] to optimize the accuracy of the model. Additionally, we have chosen the DIET (Dual Intent and Entity Transformer) Classifier [12] to carry out multi-class classification. The processing of the NLU pipeline is performed on a computer running Ubuntu 20.04. This setup includes Tensorflow (2.6.1), CUDA (11.2), and two Nvidia GPU 1080Ti graphics cards. The software stack is completed with Rasa (3.0), Node.js (17.2.0), and MongoDB (5.0.4) installations.

4.3 Training and Updating

4.3.1 Data Saving and Retrieval. The NLU pipeline is trained on each user's individual coding history throughout the coding process. In particular, every user's coded data is independently stored in the database. New proposed codes are then compared with the user's own coding history for being grouped and deduplicated. For instance, if two different sentences receive the same code from one user, they're grouped into a singular "intent" (analogous

---

[3]https://rasa.com/docs/rasa/components/#dietclassifier

[4]https://rasa.com/docs/rasa/. Rasa has previously been employed to facilitate conversations in other prototypes within the HCI field [47].

[5]https://rasa.com/docs/rasa/tuning-your-model/#configuring-tensorflow

[6]https://spacy.io/usage/models

to the "class" concept in Machine Learning or the "code" in this context) within the Rasa NLU data file: *nlu.yml.* If two similar sentences are coded with two distinct codes by one coder, it is grouped into the "example" for both "intent". Conversely, if two codes carry similar meanings but distinct expressions, they're treated as two separate "intent" for the current version of *AIcoder*.

4.3.2 Real-Time Training. The continuously updated *nlu.yml* file is fed into the NLU pipeline, triggering the automatic training of an updated NLU model. Subsequently, the trained model is immediately uploaded to the Rasa HTTP server, replacing the preceding model. The entire pipeline typically takes between 10 to 20 seconds, and this duration may increase as the volume of coding data grows. To effectively handle user requests and simulate real-time training, we establish two Rasa Open Source servers running on separate ports, utilizing a server-swapping mechanism as a buffering strategy. Specifically, users are able to solicit code suggestions from either of the two servers via HTTP. If *AIcoder* fails to receive a response from one server due to an ongoing model update process, it promptly switches to the alternate server.

5 STUDY DESIGN

We conducted a user study to assess the impact of various coding strategies on user trust and reliance in *AIcoder*. To establish different levels of granularity described in section 3.3, we set parameters for the length of the text selection and the associated code. Users were then asked to undertake qualitative coding at the sentence level, paragraph level, or with more flexible selection within a paragraph. Additionally, they were requested to summarize their codes in either a concise manner (short phrases of no more than three words), a more extended format (phrases containing four to six words), or in a freer style (phrases of mixed lengths ranging from one to six words). These specifications were derived from pilot studies conducted prior to the formal study.

Ultimately, we evaluated the model's performance (RQ1 in section 6.1); the *Decision Time* and *Coding Behavior* (RQ2 in section 6.2); the *Selecting Rate* and user reliance examination (RQ3 in section 6.3); participants' self-reported trust in *Perceived Trustworthiness* and *Perceived Helpfulness* of the system (RQ4 in section 6.4); as well as their subjective preferences (RQ5 in section 6.5).

5.1 Study Task

5.1.1 Dataset. We selected the reviews at random from the publicly accessible Yelp reviews dataset[7] for our open coding task. We chose Yelp reviews due to two main reasons. First, the content of reviews is a form of text that most people are familiar with, thus facilitating the coding process for participants without imposing significant difficulties. Second, reviews often come in short paragraphs, increasing the likelihood that a code assigned to one paragraph could also apply to another.

5.1.2 Pilot Test. In order to ascertain that our participants could complete the open coding tasks within a reasonable timeframe of 1 to 1.5 hours, we conducted a pilot study involving 6 graduate students with proficient English skills. This exercise revealed that a coding task comprising eight paragraphs (with an average of 86.8 words per paragraph) was the most suitable length for our study.

Our pilot study also uncovered several typographical and grammatical errors, along with colloquial references that could potentially hinder participants' understanding. To remedy this, we thoroughly cleaned the text before using it for our open coding task. Figure 4 depicts one of the revised reviews used in the formal coding tasks.

5.2 Independent Variables and Conditions

We implemented a split-plot design [31] to investigate the effects of two facets of qualitative coding granularity: *Text Granularity* (i.e., unit of analysis or length of text selection) and *Code Granularity* (i.e., length of code in

---

[7]https://www.yelp.com/dataset/documentation/main

> **3rd in 8 paragraphs**
> This is a so-called restaurant that doesn't do anything a restaurant should do except preparing food, the rest is left to the guest. Do you want water? Get up and go across the yard to get it. If you want a drink, go downstairs and pay in cash. Want to sit in dirty deckchairs in a dirty garden, enjoy yourself. The waiters are a little bit helpful as they bring you our food after you go to the window and pay cash for it. Kind of like New Orleans Hamburger and Seafood, but dirty and with live music (which is nice). It's a once in a lifetime experience for me... just once.

Fig. 4. A sample paragraph for the open coding tasks, extracted and preprocessed from the Yelp reviews dataset.

words). The first variable comprises three levels: *Sentence*, *Paragraph*, and *Selective*. The second variable also includes three levels: *Short Codes* (1-3 words), *Long Codes* (4-6 words), and *Mixed Codes* (1-6 words). Consequently, this study encompasses a total of $3 \times 3 = 9$ conditions (see Table 1 and Figure 5).

Table 1. Nine conditions corresponding to *Text Granularity* (i.e., unit of analysis or length of text selection) and *Code Granularity* (i.e., length of code in words).

		Text Granularity (text length)		
		Sentence (S)	*Paragraph* (P)	*Selective* (E)
Code Granularity (code length)	*Short Codes* (1-3 words) (S)	SS	SP	SE
	Long Codes (4-6 words) (L)	LS	LP	LE
	Mixed Codes (1-6 words) (M)	MS	MP	ME

Fig. 5. Nine Coding Methods.

We opted for a mixed-design approach (i.e., a split-plot design) to ensure the experiment duration remained manageable (approximately 1 hour). We designated *Code Granularity* as a between-subject variable to avoid affecting the participants' coding process, particularly their decision-making regarding labels.

Meanwhile, *Text Granularity* was counterbalanced according to appearance order across various levels, utilizing a Latin Square method [31]. For each of the three levels of *Text Granularity*, participants were asked to peruse eight selected texts and carry out an open coding task. The impact of differing text selection lengths can be evaluated by comparing conditions within the same row of Table 1. Conversely, the effect of varying code lengths can be ascertained by comparing conditions within the same column.

5.3   Participants

We conducted our study with 30 participants (12 males and 18 females, mean age = 21.9 years old). We based our participant selection on the following criteria: 1) age of 18 or above, 2) proficient English reading and writing skills, and 3) enrollment in or completion of an undergraduate programme. All participants, being novices in qualitative coding, received appropriate coding training from us prior to the formal study. Each participant received compensation for their time equivalent to 7.25 USD per hour, which aligns with the standard rate approved by our institution's IRB. Additionally, for the follow-up study conducted in section 6.3.3, we recruited 6 participants (3 females, mean age = 26.7 years old) with the same requirements.

5.4   Procedure

We partitioned the 30 participants into three groups of 10, each group assigned to propose either short, long, or mixed codes. Independently of their group, all participants underwent the *Sentence*, *Paragraph*, and *Selective* conditions. They were instructed to code each sentence, with the option to skip any that were devoid of meaning (*Sentence*); to assign one code to each paragraph (*Paragraph*); and to code text selections of any length within a paragraph, ranging from phrases to individual or multiple sentences (*Selective*).

Upon signing the consent form, participants were initially briefed on *Open Coding*. This was followed by a 15 to 20-minute training session, introducing them to qualitative coding. Subsequently, they began coding tasks under their assigned conditions. They were also given specific research questions to guide their coding process. Primarily, these queries necessitate participants to discern the customers' opinions and attitudes regarding the store or restaurant presented in the coding material.

To gain insights into users' attitudes towards AI, we implemented a think-aloud protocol during the study. This approach facilitated the observation of participants' coding processes and ensured the tasks were performed correctly. After each study, participants completed a survey and were encouraged to share their reasoning behind the given choices in the survey with the facilitator, and to compare the conditions they experienced. Additionally, we conducted a semi-structured interview at the end of the study to encourage participants to reflect on their experiences.

5.5   Dependent Variables

The study aims to investigate the extent to which users trust, rely on, and find the AI system helpful.

5.5.1   Model Performance. To assess the influence of the coding strategies on the model's performance, we employed evaluation metrics from recommendation systems [1, 56]: in both automatic[8] and human evaluations[9], we consider Precision@k and Mean Average Precision (MAP@k), where k signifies the number of suggestions; in automatic evaluations, we also apply Recall@k.

To facilitate a streamlined evaluation, we limit ourselves to the top five suggestions, denoted as $k = 5$. Our observations indicated that the trends for other values of k are similar to those found within the top five. Furthermore, to facilitate the computation of these metrics, we consider the user's finalized codes as approximately equivalent to the ground truth for each selected text segment[10].

---

[8]Automatic Evaluation: We use SentenceTransformers (https://www.sbert.net/) to calculate the similarity between the 'ground truth' and the recommendations. A recommendation is deemed relevant if the similarity score meets a predetermined threshold and is considered irrelevant if it falls short of this threshold.

[9]Human Evaluation: Echoing the automatic evaluation, two authors assigned labels to a subset of code recommendations based on their relevance, designating '0' for 'irrelevant' and '1' for 'relevant'. They convened twice to establish a shared understanding of the labeling criteria. Upon attaining an inter-rater reliability score (Cohen's kappa κ) exceeding 0.8, one author proceeded to label the remaining predictions, involving the other author for consultation on more intricate cases.

[10]Computing these metrics in the qualitative coding context requires a ground truth for each text segment. However, qualitative coding is inherently subjective and personal, leading to a lack of consensus on what constitutes a correct suggestion. Two coders might hold conflicting

Specifically, $Precision@k(u) = \frac{|rel(u) \cap rec_k(u)|}{k}$ is calculated as the proportion of relevant recommendations among the top k recommendations provided by the system; $Recall@k(u) = \frac{|rel(u) \cap rec_k(u)|}{|rel(u)|}$ is defined as the proportion of relevant recommendations within the top-k recommendations out of the total number of relevant recommendations. Likewise, mean $AP@k(u) = \frac{1}{|rec_k(u)|} \sum_{i \in rec_k(u)} \mathbb{I}(i \in rel(u)) Precision@rank(u, i)$ is calculated as the average of the Average Precision across all users and requests. This metric incorporates the order information, considering the relevance of items (indicated by $\mathbb{I}(i \in rel(u))$) at their respective ranks (denoted by $rank(u, i)$)[11].

5.5.2 Decision Time. The decision time refers to the duration that users spend on making a decision for each selection, starting from the moment they begin selecting until they finish entering the code. This metric can also serve as an indirect indicator of the difficulty of a coding task [61].

5.5.3 Coding Behavior. To enable a thorough comparison across our nine conditions, we examined users' coding behaviors from multiple perspectives, including the number of selections made, the length of selections (in words), the length of codes (in words), and the number of unique codes created. The *Coding Behavior* provides insights into coding strategies employed by participants.

5.5.4 Behavioral Trust. Previous studies have utilized user reliance [44, 59], which reflects the willingness to accept system suggestions, as an indicator for *Behavioral Trust*. Therefore, we also measure users' reliance [4, 20] by *Selecting Rate* = $\frac{\text{Total number of codes selected by users}}{\text{Total number of codes made}}$. The *Selecting Rate* represents the probability of users selecting a suggested code.

5.5.5 Perceived Trustworthiness. We assess users' *Perceived Trustworthiness* towards the code suggestions using a five-point Likert scale: 1 = Do not trust at all, 2 = Do not trust, 3 = Neutral, 4 = Relatively trust, 5 = High level of trust. We adapted our questions from prior research that examined users' trust levels towards classifiers and prediction results [45, 49, 63], including:

(1) How much do you trust the *Confidence Score* of the suggestions?
(2) How much do you trust the *Rank* of the suggestions?
(3) How much do you trust the system's ability to include your expected code (*Containing Ability*)?

To promote better understanding of these questions, we verbally illustrate the concepts of the confidence score and ranks to participants as they complete the survey. Furthermore, we provided explicit explanations for each question's intent to participants. For example, we clarified to participants that we were asking whether they believed the confidence score accurately reflected the suggestions' quality; if they viewed the suggestions' ranking as reliable; and if they thought the system was capable of producing their expected codes.

5.5.6 Perceived Helpfulness. Similarly, we include a question on *Perceived Helpfulness* of AI: "How helpful do you think the suggestions were?" Users were also required to provide responses using a five-point Likert scale.

5.5.7 Subjective Preferences. We obtained explicit consent from all participants to audio record the entire study. The recorded audio was subsequently transcribed verbatim into text format to facilitate further analysis.

---

views on one data point. Consequently, we regard each user's final code as the approximate 'ground truth' for the respective data. The overlap between each recommendation and this 'ground truth' is assessed using both automated techniques and human annotation. However, it's worth noting that this approach could have limitations and introduce measurement errors. We scrutinize these errors in subsequent sections.

[11]u is a user identificator; i is an item identificator; $rec_k(u)$ is a recommendation list for user containing top-k recommended items; $rel(u)$ is a list of relevant items for user u from the test set; $rank(u, i)$ is a position of item i in recommendation list $rec_k(u)$; $I[]$ is an indicator function.

Table 2. The model's performance was evaluated through both automatic and human evaluation. Note that the recall@5 metric was only available in the automatic evaluation. All values fall within the range of 0 to 1.

Factor1: Code Granularity	Factor2: Text Granularity	Automatic Evaluation			Human Evaluation	
		Precision@5 (M ± S.D.)	Recall@5 (M ± S.D.)	MAP@5 (M ± S.D.)	Precision@5 (M ± S.D.)	MAP@5 (M ± S.D.)
Short Codes (1-3 words)	Sentence	0.16 ± 0.18	0.30 ± 0.39	0.47 ± 0.13	0.20 ± 0.09	0.38 ± 0.15
	Paragraph	0.20 ± 0.13	0.68 ± 0.41	0.68 ± 0.23	0.21 ± 0.13	0.53 ± 0.31
	Selective	0.20 ± 0.16	0.47 ± 0.45	0.53 ± 0.12	0.21 ± 0.11	0.43 ± 0.19
Long Codes (4-6 words)	Sentence	0.16 ± 0.20	0.18 ± 0.19	0.42 ± 0.14	0.17 ± 0.08	0.37 ± 0.16
	Paragraph	0.54 ± 0.24	0.65 ± 0.20	0.72 ± 0.23	0.44 ± 0.15	0.80 ± 0.16
	Selective	0.42 ± 0.35	0.34 ± 0.29	0.57 ± 0.23	0.35 ± 0.24	0.56 ± 0.26
Mixed Codes (1-6 words)	Sentence	0.08 ± 0.10	0.20 ± 0.31	0.32 ± 0.09	0.12 ± 0.03	0.44 ± 0.11
	Paragraph	0.54 ± 0.30	0.66 ± 0.20	0.62 ± 0.24	0.21 ± 0.07	0.76 ± 0.20
	Selective	0.34 ± 0.22	0.44 ± 0.25	0.41 ± 0.11	0.15 ± 0.06	0.52 ± 0.11

5.6 Data Analysis

5.6.1 Quantitative analysis. We performed statistical analysis [43] using a mixed two-way ANOVA[12]: we used repeated measures on *Text Granularity*, taking into account the random effect of user. To account for the repeated measures design, we applied appropriate sphericity corrections (Greenhouse-Geisser) when needed, which adjusted both the reported p-values and degrees of freedom when necessary. Post-hoc comparisons were conducted using pairwise t-tests with Bonferroni correction to account for multiple comparisons.

5.6.2 Qualitative analysis. We employed thematic analysis [10] to derive themes and groupings for qualitative data. Upon becoming acquainted with the data and establishing initial codes, we organized the transcripts into cohesive themes that aligned with the content. We reviewed the transcripts and audio recordings to extract pertinent quotes corresponding to each identified theme.

6 RESULTS

6.1 RQ1: Impact on Model Performance

The performance of the model for both automatic and human evaluation is reported in Table 2. Complete details of the statistical analysis can be found in the Appendix A.1.

The results of both automatic and human evaluation suggest that *Code Granularity* does not exert a significant influence on Precision@5, Recall@5, and MAP@5 metrics.

However, **Text Granularity does significantly impact Precision@5, Recall@5, and MAP@5 metrics** ($p < .01$ **for all**). We noted a trend where **models performed consistently worse under *Sentence* than *Selective* and *Paragraph*** ($p < .05$ for most metrics). Furthermore, ***Paragraph* typically outperforms in comparison to *Selective***, with $p < .01$ for most differences. Our findings also suggest no interactions between *Code Granularity* and *Text Granularity*.

For human evaluation, we found no significant effects of *Text Granularity* on Precision@5. However, ***Text Granularity* significantly influences the MAP@5 scores** ($p < .001$). The trend persists: ***Sentence* consistently scores lower than both *Selective* and *Paragraph*** ($p < .01$ **in all comparisons**), and ***Selective* consistently lower than *Paragraph*** ($p < .001$). Significant interactions between *Code Granularity* and *Text Granularity* were observed under specific conditions. *Long Codes* or *Mixed Codes* × *Paragraph* significantly outperform *Long Codes* or *Mixed Codes* × *Sentence*, and *Mixed Codes* × *Paragraph* also significantly outperform *Mixed Codes* × *Selective*.

---

[12]https://pingouin-stats.org/generated/pingouin.mixed_anova.html

6.1.1 Summary. The model, when tasked on *Paragraph*, demonstrably surpasses both *Selective* and *Sentence* in both automatic and human evaluations. The performance boost may be attributed to the verbose text in the *Paragraph* configuration, which provides a richer data set for the classifier. However, shorter text selections might inherently convey less information, thereby possibly reducing the probability of a match between users' codes and the selected text. In contrast, longer selections could offer more context, potentially leading to more accurate model classifications. However, this remains speculative, and we seek further validation through users' subjective feedback.

6.2 RQ2: Impact on Decision Time and Coding Behavior

6.2.1 Coding Behavior. The *Coding Behavior* results are outlined in Table 3. Not all comparisons bear logical consistency—for instance, comparing the length of selections in *Paragraph* and *Sentence* conditions. Further intriguing observations surface when we examine (1) the number and length of selections for *Mixed Codes*, *Short Codes*, and *Long Codes* under the *Selective* condition and (2) the difference in code length between *Mixed Codes* vs. *Short Codes* and *Mixed Codes* vs. *Long Codes*. Specifically:

(1) Under *Selective* condition, the length of selections in *Long Codes* ($M = 29.22$) significantly surpasses that in *Mixed Codes* ($M = 12.07, p < .001$) while no difference between *Mixed Codes* ($M = 12.17$) and *Short Codes* ($M = 12.07$). The number of selections shows no significant difference.

(2) In the length of code, *Long Codes* ($M = 5.05$) is longer than *Mixed Codes* ($M = 3.37, p < .001$), and *Mixed Codes* ($M = 3.37$) is longer than *Short Codes* ($M = 2.19, p < .001$).

(3) In terms of code length, there is no significant difference observed between the codes in *Selective* ($M = 3.13$) and codes *Sentence* ($M = 3.19$).

Table 3. Summary of Coding Behavior. Among nine conditions, *Mixed Codes × Selective* is the baseline, having no selection and little code constraints and thus closely representing open coding.

Factor1: Code Granularity	Factor2: Text Granularity	Coding Behavior		
		Number of Selection[a] (M±S.D.)	Length of Selection[b] (M±S.D.)	Length of Code (M±S.D.)
Short Codes (1-3 words)	Sentence	–	–	2.06 ± 0.54
	Paragraph	–	–	2.51 ± 0.59
	Selective	29.70 ± 14.09	12.17 ± 12.37	2.00 ± 0.54
Long Codes (4-6 words)	Sentence	–	–	4.88 ± 0.92
	Paragraph	–	–	5.34 ± 0.72
	Selective	15.90 ± 7.37	29.22 ± 20.71	4.93 ± 0.93
Mixed Codes (1-6 words)	Sentence	–	–	2.63 ± 1.26
	Paragraph	–	–	5.03 ± 1.08
	Selective (baseline)	**28.30 ± 12.51**	**12.07 ± 9.71**	**2.46 ± 1.26**

[a] The number of selections for *Paragraph* is consistently 8, while for *Sentence* is consistently around 35.

[b] The selection length for *Paragraph* consistently averages around 87 words, while for *Sentence*, it typically averages around 14.6 words.

6.2.2 Decision Time. Decision Times across all conditions are shown in Figure 6.

Code Granularity. A significant main effect of *Code Granularity* on *Decision Time* was detected ($F_{(2,24)} = 11.13, p < .001$). Participants tended to spend more time formulating *Long Codes* ($M = 52.2s$) compared to *Short Codes* ($M = 34.0s, p = .014$) and *Mixed Codes* ($M = 31.2s, p < .01$).
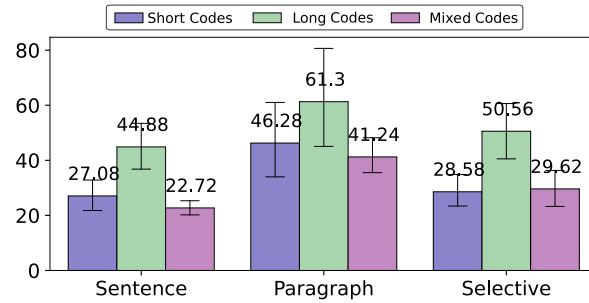
Fig. 6. Average Decision Time (Seconds). The time needed to make a decision for each selection. Final results for *Selecting Rate* and *Decision Time.* Error bars represent .95 confidence intervals.

Text Granularity. A significant main effect of *Text Granularity* on *Decision Time* was observed ($F_{(2,48)}$ = 10.13, p < .001). Generally, participants required more time to label *Paragraph* (M = 52.2s) in comparison to *Selective* (M = 35.6s, p = .023) and *Sentence* (M = 31.6s, p < .001).

Interactions. No significant interaction was detected (p = .97).

6.2.3 Summary.

Decision Time and Task Difficulty. Participants found creating *Long Codes* more challenging due to a four-word minimum, unlike the one-word minimum for *Short Codes* and *Mixed Codes*. Similarly, *Decision Time* increased with *Text Granularity*, with longer coding periods for *Paragraph*, implying greater task difficulty and time commitment for individual coding selection tasks.

Sentence ≈ Selective? Mixed Codes ≈ Short Codes? Despite certain disparities, participants demonstrated similar coding behavior across both the *Selective* and *Sentence* conditions for code length, as well as between *Mixed Codes* and *Short Codes* for number and length of selections.

Correlation between Length of Codes and Text. Crafting longer code names often necessitated larger text selections, a fact further corroborated by the text length between *Long Codes* and *Short Codes* under *Selective* condition, while the number of selections significantly decreased in longer codes conditions, while the number of selections in *Short Codes* was twice as many.

6.3 RQ3: Impact on User Reliance

In this section we examined the users' reliance on AI, we are specifically concerned about the 1) relationship between AI model performance and users' reliance. 2) whether there is a risk of overreliance that could potentially impact coding quality.

6.3.1 Selecting Rate. The *Selecting Rate* are visualized in Figure 7. Our statistical evaluation reveals that **Text Granularity** has a significant influence on **Selecting Rate** ($F_{(2,54)}$ = 15.838, p < .001), whereas *Code Granularity* does not demonstrate a main effect. Pairwise differences are detected (all p < .05): *Selective* registered the highest *Selecting Rate* (with a mean of M = 32% across conditions), followed by *Sentence* (with a mean of M = 26%), and lastly *Paragraph* (with a mean of M = 16%). Moreover, a notable interaction between *Text Granularity* and *Code Granularity* was detected ($F_{(4,54)}$ = 2.766, p = .036).

The greatest *Selecting Rate* for suggestions was accomplished with *Selective × Short Codes,* in which users chose a suggested code 40% of the time. This was succeeded by *Selective × Mixed Codes* with a *Selecting Rate* of 32%,

and *Selective × Long Codes* trailed with a lower rate of 25%. The *Selecting Rate* pattern remained consistent for *Sentence* coding, with rates fluctuating between 22% and 30%. Conversely, *Paragraph* recorded the least selection rates, which ranged from 11% to 19% overall.
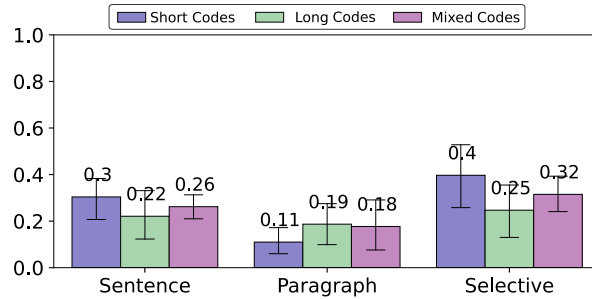

Fig. 7. *Selecting Rate* (0-1). Users' receptiveness to code suggestions produced by the system. Final results for *Selecting Rate* and *Decision Time*. Error bars represent .95 confidence intervals.

Overall, **user reliance on AI is more pronounced in the *Selective* condition**, where users are tasked with selecting only pertinent text portions—a situation that closely resembles real-world coding scenarios. On the contrary, **the *Paragraph* condition had the lowest *Selecting Rate***, particularly with *Short Codes*, likely due to the difficulty of summarizing and coding an entire paragraph with a 3-word limit. This demanding task caused both AI and participants to struggle.

6.3.2 Correlation between AI Model Performance and Users' Reliance. We have identified significant correlations ($p < .05$) for both human and automatic evaluations under two conditions: *Short Codes × Sentence* (Figure 8a), and *Long Codes × Paragraph* (Figure 8b). In these situations, users displayed an increased *Selecting Rate* as MAP@k increased. This implies that during the coding process of our study, users garnered more assistance from the system in their decision-making as the performance increased.

We should note, however, that we have used the users' final codes as approximations of 'ground truth'. A potential 'measurement error' could arise, indicating the possibility of 'overreliance'. This is because users might have accepted or made minor modifications after selecting the suggestions–even though they are not the best choice. This could conceivably lead to inflated performance metrics, subsequently resulting in an increase in the MAP@k value (see more details in the acknowledged limitation in section 9).

Given the strong correlation between *Selecting Rate* and model performance, we are concerned that this effect could be more pronounced, suggesting that some coders may have overly relied on the system under these conditions, thereby affecting the final coding quality. This concern is elevated when observing the specific *Selecting Rate* for each coder: 6 out of 30 participants demonstrated a *Selecting Rate* above 50%. When reviewing the *Selecting Rate* of these participants under the *Sentence* conditions, a similar trend emerged.

6.3.3 Comparing the Coding Results With and Without AI Assistance.

Supplementary Study. In order to validate our concern, we carried out a supplementary study. This involved 6 more participants performing coding tasks without AI assistance under conditions suspected to foster over-reliance. The procedure is identical to that of our primary study.

The experimental setup encompasses not only the two previously mentioned conditions that demonstrated a robust correlation between model performance and AI but also a condition that serves as the principal baseline (*Mixed Codes × Selective*). Additionally, we opted not to include the condition with the highest *Selecting Rate*

(a) Correlation between selection rate and MAP@5 for *Short Codes × Sentence*.

(b) Correlation between selection rate and MAP@5 for *Long Codes × Paragraph*.

Fig. 8. Significant correlations ($p < .05$) for both human and automatic evaluations.

(*Short Codes × Selective*) due to the analogous behavior observed between users for *Selective* and *Sentence*, *Mixed Codes* and *Short Codes* (see section 6.2.3). This similarity led us to anticipate that an analysis of the *Short Codes × Sentence* and *Mixed Codes × Selective* condition would probably yield results comparable to those from the *Short Codes × Selective* condition.

Results. We decided to perform a qualitative comparison between the final quality of the coding results with and without AI assistance. The results are detailed in Table 4.

We observed that codes in *Short Codes × Sentence* with AI assistance seem to have lesser code variances than coding without AI, even though their primary category is similar. For instance, 'bad food' is a code with AI assistance, while codes such as 'cold, old fries' or 'dislike burger set' serve as analogs of 'bad food' without AI assistance, albeit with more variance. When assisted by AI, users might be presented with a 'bad food' suggestion, which they may subsequently adopt instead of proposing other expressions.

The *Short Codes × Sentence* condition appears 'acceptable', with participants exhibiting a relatively commendable *Selecting Rate*. Likewise, the *Mixed Codes × Selective* condition reveals a similar change in code results. We might anticipate similar decreases in other conditions like *Short Codes × Selective*. Interestingly, the *Long Codes × Paragraph* condition seems to demonstrate a relative consistency, regardless of the presence or absence of AI.

Indeed, the decrease in code variance, to a certain extent, could be perceived as beneficial since it might result in more focused coding and reduce the effort needed to group variances. **However, it risks yielding coding outcomes that appear less substantial and somewhat superficial. This could potentially influence the discussion and creation of a codebook in subsequent stages of real qualitative analysis.**

6.4 RQ4: Impact on Perceived Trustworthiness and Helpfulness

6.4.1 Perceived Trustworthiness. Results of users' self-reported trustworthiness are depicted in Table 5 and Figure 9.

Table 4. Comparison of typical coding results for each condition (LP = *Long Codes × Paragraph*, SS = *Short Codes × Sentence*, and ME = *Mixed Codes × Selective*), both with and without the application of AI. Each cell presents codes derived from a typical user's results under the specified condition. While the "with AI" condition may yield codes with a higher selection rate, they may lack nuanced detail. In contrast, the "without AI" condition tends to generate more detailed codes.

	With AI	Without AI
LP	**Food Quality:** cheap good dessert bad breakfast decoration overall bad food try another venue good Thai food very happy meal lazy service decent food won't return **Service and Cleanliness:** self-service dirty restaurant won't visit again good food nice people recommended visit lazy service decent food won't return	**Food Quality:** cheap good location dessert bad breakfast good server ok pizza bad burger nice people good food have games tasty coconut soup and pad thai good promotion friendly people tasty food lazy service good food average pricing **Service and Cleanliness:** poor service dirty live music avail clean good service tasty reasonable price lazy service good food average pricing
SS	**Food Quality:** cheap food bad food ok food good food expensive food quality dropped **Service:** good service bad service ok service **Ambiance/Environment:** bad decoration good music good entertainment dirty **Others:** good offer	**Food Quality:** feels cheap good food, service neutral food bad food dislike burger set cold, old fries pricey pizza okay pizza good food people coconut soup pad thai same menu tried liked coconut soup coconut soup creamy good pad thai good peanuts, noodles good chicken good hot wings lots of sauce delicious sushi, affordable affordable sushi freshness and variety **Ambiance/Atmosphere:** jaded decor new orleans vibe quiet, competent chef **Service:** poor service recovery (2) decent server lack of service water not served decent service friendly staff order mixup **Recommendations and Reviews:** positive recommendation (2) mixed review **Customer Relationship:** better previous experience lost customer (mentioned twice) purchase inconvenient recent customer overall satisfied potentially lost customer **Others:** not clean beer with brother played nintendo prefer fewer herbs near hotel, convenient promo good returning for pizza large party, hibachi
ME	**Food Quality:** food tastes bad food tastes normal tasty fresh food with huge variety good place food and people **Service:** poor service good service cheap but poor food and service **Pricing:** expensive worth worth and tasty expensive and tasty tastes and feels cheap **Others:** will not try this again unhygienic quiet	**Food Quality:** cheap good price location and dessert bad burger bad fries pricey but ok pizza good coconut soup good pat thai good hot wings great sushi and reasonable price fresh with huge variety bad service good food but expensive **Service:** bad services poor service (mentioned twice) good server nice and friendly clean environment and good service great place, food and people **Attitude and others** will not come again dirty environment good location recommended 1 for 1 offer bad service good food but expensive

Code Granularity. There is a significant main effect of *Code Granularity* on users' *Perceived Trustworthiness* to *Confidence Score* ($F_{(2,27)} = 3.449, p = .046$). The pairwise comparison revealed that the *Perceived Trustworthiness* to *Confidence Score* under the *Long Codes* condition ($M = 3.37$) surpassed that under the *Mixed Codes* condition ($M = 2.67, p = .044$). No other pairwise differences were identified.

Text Granularity. There is a significant main effect of *Text Granularity* on *Perceived Trustworthiness* to *Rank* of the code suggestions ($F_{(2,54)} = 3.512, p = .037$). Pairwise comparison showed that the *Perceived Trustworthiness* to

Table 5. Summary of Values of *Perceived Trustworthiness* and *Perceived Helpfulness*. All DVs are on a Likert scale from 1 to 5.

Factor1: Code Granularity	Factor2: Text Granularity	Perceived Trustworthiness			Perceived Helpfulness (M±S.D.)
		Confidence Score (M±S.D.)	Rank (M±S.D.)	Containing Ability (M±S.D.)	
Short Codes (1-3 words)	Sentence	2.80 ± 0.92	3.10 ± 1.37	3.00 ± 1.25	3.60 ± 1.07
	Paragraph	2.40 ± 0.97	3.10 ± 1.29	2.20 ± 1.34	2.30 ± 0.95
	Selective	3.10 ± 0.88	3.80 ± 0.92	3.30 ± 1.16	3.70 ± 1.16
Long Codes (4-6 words)	Sentence	3.30 ± 0.94	3.10 ± 0.99	3.60 ± 0.97	3.80 ± 1.32
	Paragraph	3.40 ± 0.84	3.60 ± 0.84	3.40 ± 1.17	4.30 ± 1.16
	Selective	3.40 ± 1.17	3.60 ± 0.96	3.60 ± 0.84	3.80 ± 1.31
Mix Codes (1-6 words)	Sentence	2.40 ± 1.35	2.30 ± 0.95	2.40 ± 1.35	2.70 ± 1.16
	Paragraph	3.10 ± 0.88	3.00 ± 1.05	3.40 ± 0.97	3.50 ± 0.71
	Selective (baseline)	**2.50 ± 0.97**	**3.10 ± 0.99**	**3.20 ± 1.14**	**1.40 ± 0.97**

Rank appeared to be superior in *Selective* ($M = 3.50$) compared to *Sentence* ($M = 2.83, p < .01$). No other main effects were discerned.

Interaction effects. No interaction effects were detected on *Perceived Trustworthiness*.

6.4.2 Perceived Helpfulness.

Code Granularity. A significant main effect of *Code Granularity* on *Perceived Helpfulness* were observed ($F_{(2,27)} = 9.789, p < .001$). The system's suggestions were deemed more helpful by users in the *Long Codes* condition ($M = 3.97$) than those in the *Mixed Codes* condition ($M = 2.53, p < .001$). Likewise, in the *Short Codes* condition, the system was perceived as more helpful ($M = 3.20$) than in the *Mixed Codes* condition ($M = 2.53, p = .049$).

Text Granularity. No significant main effect on *Perceived Helpfulness* was discerned.

Interaction effects. A noteworthy interaction was detected between two factors on *Perceived Helpfulness* of the system ($F_{(4,54)} = 7.94, p < .001$). Particularly, under the *Mixed Codes* condition, the system was rated significantly more helpful in the *Paragraph* condition compared to the *Selective* condition ($p < .001$).

For descriptive statistics, the mean *Perceived Helpfulness* scores exceeds 3 (refer to Figure 9), albeit experiencing slight reductions under particular conditions such as *Mixed Codes × Selective*, *Mixed Codes × Sentence*, and *Short Codes × Paragraph*. We also observed that pairing *Long Codes* with a high level of *Text Granularity* (*Paragraph*) resulted in the highest mean *Perceived Helpfulness* (4.3/5), significantly surpassing the scores in any of the other eight conditions. Unexpectedly, the baseline condition (*Mixed Codes × Selective*) results in the lowest *Perceived Helpfulness*. Moreover, when *Short Codes* is paired with *Paragraph* coding, it results in significantly diminished *Perceived Helpfulness*, with users rating the system as not helpful. We delve deeper into this phenomenon from the perspective of task difficulty in Section 7.

6.5 RQ5: Impact on Subjective Preferences

In this section, we encapsulate the feedback conveyed by participants during and subsequent to the study.

6.5.1 User Preferred Selective.
Greater control over the selection enabled participants to receive more accurate suggestions, which subsequently motivated them to choose suggestions more frequently. This is evidenced by the participant comment: *"Because I can adjust the selection, then I think the way the numbers (confidence score) work...sometimes the one on the top is the one I want."* (P26, *Mixed Codes* and *Selective*).

Even though the length of text selections was similar between *Selective* and *Sentence*, participants showed a preference for *Selective*, as articulated by P18: *"The main difference is that for the sentence one, some sentences don't*
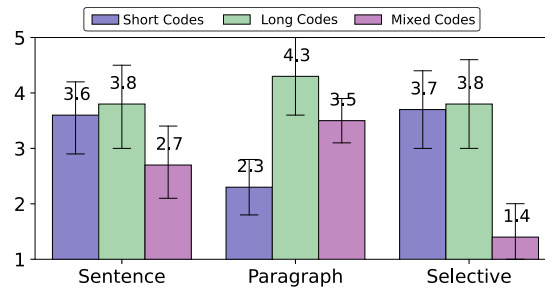
Fig. 9. User's *Perceived Helpfulness* of code suggestions. Error bars show .95 confidence intervals. Y-axis represents 1-5 Likert score, where 1 represents a complete lack of helpfulness and 5 is the highest level of helpfulness.

have meaning. But for selective, I could group the sentences with the same meaning together under one topic." (*Long Codes* and *Selective*).

6.5.2 Imperfect AI Suggestions Still Contribute Value. In many instances, participants found the suggested codes were close to their original ideas: *"I find it relatively helpful. The recommended codes bear some similarity to what I had in mind, so I don't have to ponder excessively."* (P20, *Long Codes* and *Paragraph*).

Whereas, even if the suggestions didn't always precisely align with the participants' requirements, they were still viewed as helpful. The suggested codes from the list inspired participants to combine existing codes to generate new ones and refine their own. The system's feature that allowed users to modify suggested codes was particularly appreciated:

"I think they [suggestions] are quite helpful. They kind of give you a hint about what you could write for the keywords or summaries." (P28, *Mixed Codes* and *Sentence*).

6.5.3 Code Suggestions Promote Consistency. Several participants highlighted that the suggestions aided them in maintaining consistency throughout the coding process:

"The recommendation list seems somewhat helpful because it enables me to apply a consistent metric when assessing these text streams. As a result, I can establish a bit more consistency between the texts as I formulate my codes." (P22, *Mixed Codes* and *Sentence*).

6.5.4 Too Long Text Selections (Paragraph) Presents Challenges. Overall, participants indicated that AI would need to bolster its performance to meet their expectations. Specifically, a notable challenge inherent in AI is its struggle to capture nuanced information within the text:

"The system failed to capture the context of sentences within the paragraph. At times, sentences were unrelated to one another - one might discuss good service while another addressed food, indicating different contexts within each review. Consequently, the system couldn't discern the nuances of individual sentences and provide accurate confidence scores." (P39, *Mixed Codes* and *Sentence*).

7 DISCUSSION

Our discussion first delves into an examination of task difficulty, outlining how various conditions were deemed more challenging than others. Following this, we unpack the diverse elements of trust between humans and *AIQCs* under evaluation. Lastly, we traverse through the significance of differing granularity conditions, especially in relation to their relevance in more realistic qualitative analysis scenarios.

7.1 Task Difficulty Across Conditions for *Open Coding*

7.1.1 Qualitative Open Coding: A Series of Distinct Tasks Rather than a Singular Whole. In essence, our nine conditions simulate various levels of difficulty associated with *Open Coding* tasks. Our findings advance a nuanced understanding, positing that coding tasks can differ based on their intrinsic difficulty or the effort demanded. Some tasks may boost the model's performance, while others might hinder it. This viewpoint diverges from prior research in this domain, which might have predominantly treated *Open Coding* as a uniform, undifferentiated task, intending to devise a solitary method to facilitate it. This view has overlooked the inherent complexity of subjective tasks like qualitative coding, a scenario where human-AI interaction could play a pivotal role.

7.1.2 Challenging Paragraph Conditions. In *Paragraph*, the units of text to be coded were longer compared to those in the *Sentence* and *Selective* conditions. Moreover, based on subjective feedback, *Paragraph* might have included various contradictory nuanced information. Therefore, participants needed more time to decide on a code, leading us to infer that the coding task under the *Paragraph* conditions was relatively more challenging.

However, according to the model performance data, an increase in context and text selection seems to enable the model to get higher precision. Conversely, a decrease in the text selection did not yield the same level of model performance.

At first glance, they may seem contradictory to each other. However, a paragraph may have a higher probability of matching the code, but it may also contain extraneous and even contradictory information not included in the "highly matching code". Consequently, despite the possibility of paragraphs yielding higher model performance scores, they present a substantial challenge to both users and AI, particularly due to the nuanced information they may encapsulate.

7.1.3 The Complexity of Long Codes Compared to Short Codes and Mixed Codes. The *Long Codes* conditions seemingly posed greater challenges for participants due to the requirement of a minimum number of words for each code, as indicated by the extended decision-making times. Conversely, participants found the *Short Codes* conditions less strenuous as they closely mirrored conventional coding scenarios, with code lengths similar to those in the *Mixed Codes*.

7.2 Trust Discrepancies Due to Varied Task Difficulties

7.2.1 Higher Behavioral Trust for Simpler Tasks. In terms of *Text Granularity*, our data reveal that users exhibit greater *Behavioral Trust*, or reliance on AI, in simpler tasks at the *Selective* and *Sentence* levels, as indicated by a 26%+ suggestion selection rate. This contrasts with the harder *Paragraph* tasks, which only saw an average selection rate of 16%.

Significant individual differences also emerged. For instance, in *Selective* and *Sentence* conditions, certain participants (P14 for *Selective*, P7 for *Sentence*) ignored all suggestions, while others had high selection rates (73% for P10 in *Selective*, 53% for P17 in *Sentence*). For the tougher *Paragraph* conditions, an overwhelming 11 participants completely disregarded the system's suggestions, with the peak selection rate merely reaching 38% (P25, P27, P28).

7.2.2 Contrasting Behavioral Trust and Perceived Helpfulness in Complex Tasks. Interestingly, while *Behavioral Trust* was at its nadir in the *Paragraph* setting, *Perceived Helpfulness* was substantially high, and *Perceived Trustworthiness* was also considerable, especially for more complex tasks with longer AI suggestions. In particular, during tasks with longer codes (*Long Codes × Paragraph* and *Mixed Codes × Paragraph*), users, while finding individual suggestions inadequate for selection (thus the low *Selecting Rate*), still referenced them or made minor adjustments to construct their codes. This added flexibility enhanced their *Perceived Helpfulness*, especially in the challenging task of *Long Codes × Paragraph*, scoring 4.3/5 in *Perceived Helpfulness*.

7.3 Over- and under-reliance on AIQCs

As discussed, different coding tasks resulted in varying reliance on the system. However, *AIQCs* should strike a balance between user exploration and AI assistance, without promoting either an excessive reliance or insufficient use of AI suggestions. Over-reliance can lead to shallow coding, while under-reliance may result in missed opportunities for valuable AI assistance. As such, striking the right balance is crucial for the effective use of AI in qualitative coding.

7.3.1 Reasons for Under-reliance. The low *Selecting Rate* for *Paragraph* tasks is primarily due to fewer coding units, resulting in fewer data points for model training. Typically, participants would only choose from the last few suggestions, with a total selection count of approximately 8, in comparison to around 35 in *Sentence* and 20+ in *Selective* tasks. Thus, we anticipate that with more data points to train the model, the system reliance would increase.

For those scenarios where data points may not significantly increase, enabling users to edit their codes post-selection could offer indirect assistance (through the process of selection and then editing). Although these improvements might not be prominently reflected in *Selecting Rate*, they could elevate users' subjective experience, potentially bolster the system's trustworthiness, and encourage users to fully exploit the system [6].

7.3.2 Over-reliance Risk. As detailed in section 6.3.2 and 6.3.3, reliance could ostensibly reduce human effort by enabling users to re-utilize previous codes, causing a more focused coding. However, it also carries an over-reliance risk. Over-reliance might disrupt the delicate balance between focused coding and the generation of diverse coding outcomes, which could narrow the scope of interpretation and potentially reduce the depth and breadth of the coding process. Therefore, ensuring a balance between AI assistance and human input is key to maximizing both the efficiency and depth of qualitative coding. This balance should never be underestimated in the design of a truly trustworthy *AIQCs*, as opposed to a system that deceives users' trust without meriting it [6].

7.4 Optimal Code Granularity Varies Between Users and AI

Participants usually generate shorter codes when possible. The *Mixed Codes*, which allows them to create more specific and longer codes, resembles real-life open coding tasks more closely. The similar code lengths in *Short Codes* and *Mixed Codes* under both *Selective* and *Sentence* conditions imply that participants aim to minimize their codes' length when given the option in the given study. Hence, *Short Codes* or *Mixed Codes* can be considered optimal code granularity for users.

Conversely, while users prefer to add shorter codes, they anticipate longer code suggestions from AI. This is evident in the consistently higher perceived helpfulness of *Long Codes* over other *Text Granularity* conditions. We infer that longer AI suggestions enable more expressiveness, thereby reducing potential misinterpretations between users and AI.

Moreover, a discrepancy exists between human and AI preferences when it comes to text selection for coding. Participants generally favored labeling shorter selections that accommodated shorter codes, as indicated by the similar selection lengths in the *Short Codes × Selective* and *Mixed Codes × Selective* cases, where users selected only the critical single semantic elements for coding. On the other hand, AI favored a more comprehensive context for accurate code prediction, thus creating a divergence between human and AI inclinations. In particular, the *Short Codes × Paragraph* condition exemplified a considerable mismatch. Although it offers the optimal code length for users, the restriction of a three-word code for an extensive text led to a disconnect between the text and code, significantly increasing the task difficulty. This resulted in the lowest *Selecting Rate* of 11% among users and lower *Perceived Helpfulness* than neutral (2.3/5).

In addition, users seemed to favor uniform AI suggestions, as indicated by their perception of *Mixed Codes* suggestions as less helpful than both *Long Codes* and *Short Codes*. The mix of long and short codes in the suggestion

list under *Mixed Codes* might have led to information overload, making it challenging for users to locate useful information. Moreover, while the added flexibility, notably in *Selective* codes, could theoretically benefit the users, it seemed to inadvertently decrease the user's perceived helpfulness of the suggestions in *Mixed Codes × Selective*.

7.5 Coding Strategies in Real Life

7.5.1 Selective is Best for Coding. Notably, we observed the highest levels of *Behavioral Trust* in the *Selective* coding conditions. *Selective* coding most accurately emulates how a single researcher might begin to navigate data in a real-life scenario. They would select the most pertinent phrases and then generate a suitable label. In practice, it is crucial to utilize various coding levels, alternating perspectives, and varying depths of understanding to produce a more comprehensive and diverse range of codes.

7.5.2 Sentence for Collaborative Coding. In fact, *Selective* and *Sentence* coding scenarios share several similarities, notwithstanding certain notable differences. The variance between these two conditions is significantly less than that between them and the *Paragraph* condition. While *Selective* may generally be the go-to granularity for coding, *Sentence* level coding can be particularly beneficial for collaborative coding, where consistency between multiple users is required. This is especially important when computing inter-rater reliability scores, as it requires a straightforward, unambiguous text selection unit.

7.5.3 Paragraph for Summarizing Long Texts. Paragraph may still prove valuable for users attempting to summarize lengthy texts in real coding scenarios. Opting for a *Paragraph* approach assists in distilling entire pages into a few concise labels.

8 IMPLICATIONS FOR DESIGN

We propose several guidelines to cultivate appropriate reliance and foster a productive human-AI collaboration within the context of *AIQCs*.

8.1 Fostering Trustworthiness during Under-reliance on *AIQCs*

8.1.1 Offering Extensive and Modifiable Suggestions. We observed that while users generally found less difficulty in creating *Short Codes* and *Mixed Codes*, *Long Codes* suggestions seem to be perceived as more beneficial and trustworthy. The utility of longer suggestions stems from their capacity to convey a wealth of information, thereby minimizing ambiguity and potentially delivering deeper meaning. This extensive nature also enables users to refine their code by editing suggestions, tailoring them to their unique requirements. This active participation makes users feel more in control, which could result in increased trust and system usage.

8.1.2 Exploiting Larger Training Datasets. We noted that some participants did not utilize AI suggestions during the *Paragraph* tasks. This lack of use could be attributed to the reduced quantity and quality of the data used for training, resulting in initially subpar AI suggestions.

To address this concern, we propose the application of data augmentation techniques[13], generating additional training data. Furthermore, if feasible, the integration of data from diverse users and sources could be beneficial for open coding. This recommendation aligns with the current trend towards a data-centric approach, as advocated in recent literature [21, 39, 60].

8.1.3 Facilitating Open Coding Through Multifaceted Models. We further recommend utilizing multiple models to generate code outputs from diverse perspectives. Rather than exclusively relying on text classification or topic modeling, we advocate for considering and integrating other methodologies, such as Generative AI like GPT [14].

---

[13]https://www.tensorflow.org/tutorials/images/data_augmentation
[14]https://atlasti.com/ai-coding-powered-by-openai, https://openai.com/chatgpt

By doing so, users can construct their codes under a wider umbrella of system assistance, thereby enabling more informed decision-making [24, 28]. Moreover, the system could offer suggestions inspired by codes from other users, thus presenting an alternate view of the data.

8.2 Mitigating Over-reliance to Prevent Shallow Codes

We've recognized the potential for over-reliance in certain situations. At times, AI that lacks sufficient trustworthiness could deceive users into considering it 'trustworthy' [6], leading to excessive reliance. Below, we delve into several specific design strategies to mitigate this issue.

8.2.1 Implementing a Delay in Suggestions Display upon Selection. The system could be designed to deliberately delay the display of suggestions or only present codes upon a user's request, ensuring they appear specifically when a user struggles to formulate a code [11]. This feature would afford the user sufficient time to contemplate an initial code, and subsequently ensure that the displayed suggestions align effectively with their requirements.

8.2.2 Providing Explanations for AI Suggestions. A promising strategy might be to present explanations alongside the code suggestions [57]. For example, by displaying the original data from which the suggestions are derived, coders can compare and ascertain the appropriateness of coding the current data under a specific code. This approach not only encourages deeper thinking but also fosters appropriate reliance on the system.

9 LIMITATIONS AND FUTURE WORK

This work has limitations. First, understanding the accuracy and overall performance of the model is crucial for gauging the system's effectiveness under varying conditions. Ideally, each text segment should have a corresponding ground truth value (such as 0 or 1) against which we can compare system recommendations to evaluate model performance.

To approximate this, we have considered each user's final code as a proximate 'ground truth' for the specific text segment. This approximation is based on two assumptions: 1) the system merely plays an assisting role while the user remains the ultimate decision-maker; 2) all recommendations are derived from users' own coding history, enabling them to fully comprehend these suggestions. They can then decide whether to accept, modify, or reject these suggestions in order to make a final decision. Therefore, their assigned code for a text segment can thus be considered as a close 'ground truth' representation for various users.

Nevertheless, we understand that our approach only provides an estimation of the model's performance and may introduce **measurement errors**: 1) using the user's final code as 'ground truth' presumes that the user's decisions are always accurate and ideal. However, users can make mistakes or demonstrate biases in their coding decisions. In such scenarios, the model's performance evaluation for a given text segment may be flawed, as the 'ground truth' itself might not be correct; 2) interpreting the final user decision as 'ground truth' could potentially inflate the model's performance metrics. Users often accept or make minor modifications to system suggestions, but this does not necessarily signify the model's recommendations were entirely accurate or the best available option. For example, in instances where users frequently adopt the model's recommendations despite them being merely "not bad" as opposed to the best, the performance assessment could be artificially elevated. As such, the model may appear to have a higher performance score, not necessarily because it offers the best suggestions, but because users tend to agree with its recommendations. This inflated performance metric could potentially misrepresent the model's true capacity to deliver optimal solutions across diverse contexts and user behaviors.

While we have accounted for this measurement error in our interpretation, future research could further investigate better ways to evaluate model performance, or establish a more suitable ground truth for subjective tasks such as qualitative coding.

Moreover, our choice to focus on codes of varying abstraction levels and specificity stems primarily from their representation of different user coding habits has been stated in Section 3.3. In particular, to emulate these different levels of abstraction and interpretation, we opted for a simplistic albeit imperfect method for user operation. Codes of three words or less represent concise coding (short codes), those between four to six words signify verbose coding (long codes), and codes ranging from one to six words (mixed codes) represent natural coding. We selected these parameters for the study setup based on pilot tests conducted on our own materials prior to the formal study. However, we acknowledge that this classification has its shortcomings. For instance, the specific length of the codes is intimately linked to the domain of the coding material, and the delineation of codes across different levels of the factor remains unclear. Looking forward, it would be promising to extend these results to various types of content, with the goal of gaining a more comprehensive understanding of the specific assistance and suggestions users truly need.

Furthermore, there are also some limitations when assessing user trust (*Perceived Trustworthiness*) in *AIQCs*. For instance, users may struggle to differentiate their specific emotions and levels of trust towards individual components of the system (confidence score, rank, and containing ability), thereby influencing the overall evaluation of *Perceived Trustworthiness*. To address this, future research should invest more effort into developing more precise measures for trust evaluation.

Additionally, the selected parameters of limitation (1-3 words, 4-6 words, etc.) used for coding are, while simplistic, imperfect representations of user operations, mirroring the range from concise to lengthy and natural coding habits. The specific values, however, could vary significantly based on the coding material. Moreover, it's essential to motivate participants to execute tasks with greater efficiency, thereby achieving a more precise measure of decision-making time. Future studies should further investigate these aspects, aiming to devise more general strategies for controlling and managing human-AI interaction habits.

Overall, our primary objective in this work is to appeal to developers and researchers, underlining the importance of developing trustworthy *AIQCs* that fosters robust human-AI collaboration by taking into account the unique dynamics of human-AI interaction within qualitative coding. It is critical to not only integrate advanced technologies into this domain but also to view *Open Coding* as a collection of different subtasks. Therefore, the design of various tools should aim to support the nuanced and varied coding tasks inherent within *Open Coding*. Furthermore, the potential risks for under-utilization (under-reliance) and over-reliance should be considered, as the former could result in the system being under-utilized, and the latter might lead to less insightful coding outcomes.

10 CONCLUSION

Issues concerning trust between humans and AI in *AIQCs* have been identified, but the exploration has remained limited. In this work, we explored how *Code* and *Text Granularity* could influence user trust and reliance in *AIQCs* by conducting a split-plot design study with 30 participants and a follow-up study with 6 participants. Our study highlighted that *Open Coding*, due to its unique human-AI interaction dynamics, should be approached as a composite of various subtasks. Each of these subtasks necessitates a tailored design. Our findings also indicate trust discrepancies stemming from varied subtask difficulties and illuminate the problems of over-reliance and under-reliance existing in different conditions. These results form a foundation for future research on the user trust, reliance, and utility of *AIQCs*.

REFERENCES

[1] [n. d.]. Metrics. Retrieved April, 2023 from https://darel13712.github.io/rs_metrics/metrics/
[2] Saleema Amershi, Maya Cakmak, William Bradley Knox, and Todd Kulesza. 2014. Power to the people: The role of humans in interactive machine learning. *Ai Magazine* 35, 4 (2014), 105–120.

[3] Mohammad Amiryousefi, Masumeh Sadat Seyyedrezaei, Ana Gimeno-Sanz, and Manssor Tavakoli. 2021. Impact of Etherpad-based Collaborative Writing Instruction on EFL Learners' Writing Performance, Writing Self-efficacy, and Attribution: A Mixed-Method Approach. *Two Quarterly Journal of English Language Teaching and Learning University of Tabriz* 13, 28 (2021), 19–37.

[4] Zahra Ashktorab, Michael Desmond, Josh Andres, Michael Muller, Narendra Nath Joshi, Michelle Brachman, Aabhas Sharma, Kristina Brimijoin, Qian Pan, Christine T Wolf, et al. 2021. AI-Assisted Human Labeling: Batching for Efficiency without Overreliance. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW1 (2021), 1–27.

[5] Tita Alissa Bach, Amna Khan, Harry Hallock, Gabriela Beltrão, and Sonia Sousa. 2022. A Systematic Literature Review of User Trust in AI-Enabled Systems: An HCI Perspective. *International Journal of Human–Computer Interaction* (2022), 1–16.

[6] NIKOLA BANOVIC, ZHUORAN YANG, ADITYA RAMESH, and ALICE LIU. 2023. Being Trustworthy is Not Enough: How Untrustworthy Artificial Intelligence (AI) Can Deceive the End-Users and Gain Their Trust. (2023).

[7] Eric PS Baumer, David Mimno, Shion Guha, Emily Quan, and Geri K Gay. 2017. Comparing grounded theory and topic modeling: Extreme divergence or unlikely convergence? *Journal of the Association for Information Science and Technology* 68, 6 (2017), 1397–1410.

[8] Sarah Bebermeier and Denise Kerkhoff. 2019. Use and Impact of the Open Source Online Editor Etherpad in a Psychology Students' Statistics Class. *Psychology Teaching Review* 25, 2 (2019), 30–38.

[9] Michelle Brachman, Zahra Ashktorab, Michael Desmond, Evelyn Duesterwald, Casey Dugan, Narendra Nath Joshi, Qian Pan, and Aabhas Sharma. 2022. Reliance and Automation for Human-AI Collaborative Data Labeling Conflict Resolution. *Proceedings of the ACM on Human-Computer Interaction* 6, CSCW2 (2022), 1–27.

[10] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.

[11] Zana Buçinca, Maja Barbara Malaya, and Krzysztof Z Gajos. 2021. To trust or to think: cognitive forcing functions can reduce overreliance on AI in AI-assisted decision-making. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW1 (2021), 1–21.

[12] Tanja Bunk, Daksh Varshneya, Vladimir Vlasov, and Alan Nichol. 2020. Diet: Lightweight language understanding for dialogue systems. *arXiv preprint arXiv:2004.09936* (2020).

[13] Shiye Cao and Chien-Ming Huang. 2022. Understanding User Reliance on AI in Assisted Decision-Making. *Proceedings of the ACM on Human-Computer Interaction* 6, CSCW2 (2022), 1–23.

[14] Kathy Charmaz. 2014. *Constructing grounded theory.* sage.

[15] Nan-Chen Chen, Margaret Drouhard, Rafal Kocielnik, Jina Suh, and Cecilia R Aragon. 2018. Using machine learning to support qualitative coding in social science: Shifting the focus to ambiguity. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 8, 2 (2018), 1–20.

[16] Nan-chen Chen, Rafal Kocielnik, Margaret Drouhard, Vanessa Peña-Araya, Jina Suh, Keting Cen, Xiangyi Zheng, and Cecilia R Aragon. 2016. Challenges of applying machine learning to qualitative coding. In *ACM SIGCHI Workshop on Human-Centered Machine Learning.*

[17] Juliet Corbin and Anselm Strauss. 2014. *Basics of qualitative research: Techniques and procedures for developing grounded theory.* Sage publications.

[18] Kevin Crowston, Xiaozhong Liu, and Eileen E Allen. 2010. Machine learning and rule-based automated coding of qualitative data. *proceedings of the American Society for Information Science and Technology* 47, 1 (2010), 1–2.

[19] Jessica T DeCuir-Gunby, Patricia L Marshall, and Allison W McCulloch. 2011. Developing and using a codebook for the analysis of interview data: An example from a professional development research project. *Field methods* 23, 2 (2011), 136–155.

[20] Mary T Dzindolet, Scott A Peterson, Regina A Pomranky, Linda G Pierce, and Hall P Beck. 2003. The role of trust in automation reliance. *International journal of human-computer studies* 58, 6 (2003), 697–718.

[21] Sabri Eyuboglu, Bojan Karlaš, Christopher Ré, Ce Zhang, and James Zou. 2022. dcbench: a benchmark for data-centric AI systems. In *Proceedings of the Sixth Workshop on Data Management for End-To-End Machine Learning.* 1–4.

[22] Noura Farra, Elie Challita, Rawad Abou Assi, and Hazem Hajj. 2010. Sentence-level and document-level sentiment mining for arabic texts. In *2010 IEEE international conference on data mining workshops.* IEEE, 1114–1119.

[23] Cristian Felix, Aritra Dasgupta, and Enrico Bertini. 2018. The exploratory labeling assistant: Mixed-initiative label curation with large document collections. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology.* 153–164.

[24] Jessica L Feuston and Jed R Brubaker. 2021. Putting Tools in Their Place: The Role of Time and Perspective in Human-AI Collaboration for Qualitative Analysis. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW2 (2021), 1–25.

[25] 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.* 1–19.

[26] Max Goldman, Greg Little, and Robert C Miller. 2011. Real-time collaborative coding in a web IDE. In *Proceedings of the 24th annual ACM symposium on User interface software and technology.* 155–164.

[27] Matt-Heun Hong, Lauren A Marsh, Jessica L Feuston, Janet Ruppert, Jed R Brubaker, and Danielle Albers Szafir. 2022. Scholastic: Graphical Human-AI Collaboration for Inductive and Interpretive Text Analysis. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology.* 1–12.

[28] Jialun Aaron Jiang, Kandrea Wade, Casey Fiesler, and Jed R Brubaker. 2021. Supporting serendipity: Opportunities and challenges for Human-AI Collaboration in qualitative analysis. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW1 (2021), 1–23.

[29] Andreas Kaufmann, Ann Barcomb, and Dirk Riehle. 2020. Supporting Interview Analysis with Autocoding.. In *HICSS*. 1–10.

[30] Rafal Kocielnik, Saleema Amershi, and Paul N Bennett. 2019. Will you accept an imperfect ai? exploring designs for adjusting end-user expectations of ai systems. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–14.

[31] Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hochheiser. 2017. *Research methods in human-computer interaction*. Morgan Kaufmann.

[32] John D Lee and Katrina A See. 2004. Trust in automation: Designing for appropriate reliance. *Human factors* 46, 1 (2004), 50–80.

[33] William Leeson, Adam Resnick, Daniel Alexander, and John Rovers. 2019. Natural Language Processing (NLP) in qualitative public health research: a proof of concept study. *International Journal of Qualitative Methods* 18 (2019), 1609406919887021.

[34] Mengxiang Li, Liqiang Huang, Chuan-Hoo Tan, and Kwok-Kee Wei. 2013. Helpfulness of online product reviews as seen by consumers: Source and content features. *International Journal of Electronic Commerce* 17, 4 (2013), 101–136.

[35] Jasy Suet Yan Liew, Nancy McCracken, Shichun Zhou, and Kevin Crowston. 2014. Optimizing features in active machine learning for complex qualitative content analysis. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*. 44–48.

[36] Britt-Marie Lindgren, Berit Lundman, and Ulla H Graneheim. 2020. Abstraction and interpretation during the qualitative content analysis process. *International journal of nursing studies* 108 (2020), 103632.

[37] Megh Marathe and Kentaro Toyama. 2018. Semi-automated coding for qualitative research: A user-centered inquiry and initial prototypes. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–12.

[38] Roger C Mayer, James H Davis, and F David Schoorman. 1995. An integrative model of organizational trust. *Academy of management review* 20, 3 (1995), 709–734.

[39] Mohammad Motamedi, Nikolay Sakharnykh, and Tim Kaldewey. 2021. A data-centric approach for training deep neural networks with less data. *arXiv preprint arXiv:2110.03613* (2021).

[40] Michael Muller, Shion Guha, Eric PS Baumer, David Mimno, and N Sadat Shami. 2016. Machine learning and grounded theory method: convergence, divergence, and combination. In *Proceedings of the 19th international conference on supporting group work*. 3–8.

[41] Laura K Nelson. 2020. Computational grounded theory: A methodological framework. *Sociological Methods & Research* 49, 1 (2020), 3–42.

[42] Ha Nguyen, June Ahn, Ashlee Belgrave, Jiwon Lee, Lora Cawelti, Ha Eun Kim, Yenda Prado, Rossella Santagata, and Adriana Villavicencio. 2021. Establishing trustworthiness through algorithmic approaches to qualitative research. In *International Conference on Quantitative Ethnography*. Springer, 47–61.

[43] Geoff Norman. 2010. Likert scales, levels of measurement and the "laws" of statistics. *Advances in health sciences education* 15, 5 (2010), 625–632.

[44] Andrea Papenmeier, Dagmar Kern, Gwenn Englebienne, and Christin Seifert. 2022. It's Complicated: The Relationship between User Trust, Model Accuracy and Explanations in AI. *ACM Transactions on Computer-Human Interaction (TOCHI)* 29, 4 (2022), 1–33.

[45] Andrea Papenmeier, Dagmar Kern, Daniel Hienert, Yvonne Kammerer, and Christin Seifert. 2022. How Accurate Does It Feel?–Human Perception of Different Types of Classification Mistakes. In *CHI Conference on Human Factors in Computing Systems*. 1–13.

[46] Pablo Paredes, Ana Rufino Ferreira, Cory Schillaci, Gene Yoo, Pierre Karashchuk, Dennis Xing, Coye Cheshire, and John Canny. 2017. Inquire: Large-scale early insight discovery for qualitative research. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. 1562–1575.

[47] David Porfirio, Evan Fisher, Allison Sauppé, Aws Albarghouthi, and Bilge Mutlu. 2019. Bodystorming human-robot interactions. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 479–491.

[48] Li Qin and Sue Kong. 2015. Perceived helpfulness, perceived trustworthiness, and their impact upon social commerce users' intention to seek shopping recommendations. *Journal of Internet Commerce* 14, 4 (2015), 492–508.

[49] Amy Rechkemmer and Ming Yin. 2022. When Confidence Meets Accuracy: Exploring the Effects of Multiple Performance Indicators on Trust in Machine Learning Models. In *CHI Conference on Human Factors in Computing Systems*. 1–14.

[50] K Andrew R Richards and Michael A Hemphill. 2018. A practical guide to collaborative qualitative data analysis. *Journal of Teaching in Physical Education* 37, 2 (2018), 225–231.

[51] Tim Rietz and Alexander Maedche. 2021. Cody: An AI-based system to semi-automate coding for qualitative research. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–14.

[52] Johnny Saldaña. 2021. The coding manual for qualitative researchers. *The coding manual for qualitative researchers* (2021), 1–440.

[53] Nicolas Scharowski, Sebastian AC Perrig, Nick von Felten, and Florian Brühlmann. 2022. Trust and Reliance in XAI–Distinguishing Between Attitudinal and Behavioral Measures. *arXiv preprint arXiv:2203.12318* (2022).

[54] E Straws and J Korbin. 2015. Basics of qualitative research: Techniques and procedures for grounded theory (E. Afshar, Trans.). *Tehran: Nei publication.[In Persian]* (2015).

[55] Simone Stumpf, Vidya Rajaram, Lida Li, Weng-Keen Wong, Margaret Burnett, Thomas Dietterich, Erin Sullivan, and Jonathan Herlocker. 2009. Interacting meaningfully with machine learning systems: Three experiments. *International journal of human-computer studies* 67,

8 (2009), 639–662.

[56] Yan-Martin Tamm, Rinchin Damdinov, and Alexey Vasilev. 2021. Quality metrics in recommender systems: Do we calculate metrics consistently?. In *Proceedings of the 15th ACM Conference on Recommender Systems*. 708–713.

[57] Helena Vasconcelos, Matthew Jörke, Madeleine Grunde-McLaughlin, Tobias Gerstenberg, Michael S Bernstein, and Ranjay Krishna. 2023. Explanations can reduce overreliance on ai systems during decision-making. *Proceedings of the ACM on Human-Computer Interaction* 7, CSCW1 (2023), 1–38.

[58] Oleksandra Vereschak, Gilles Bailly, and Baptiste Caramiaux. 2021. How to evaluate trust in AI-assisted decision making? A survey of empirical methodologies. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW2 (2021), 1–39.

[59] Eric S Vorm. 2018. Assessing demand for transparency in intelligent systems using machine learning. In *2018 Innovations in Intelligent Systems and Applications (INISTA)*. IEEE, 1–7.

[60] Steven Euijong Whang, Yuji Roh, Hwanjun Song, and Jae-Gil Lee. 2021. Data Collection and Quality Challenges in Deep Learning: A Data-Centric AI Perspective. *arXiv preprint arXiv:2112.06409* (2021).

[61] George Wright and Peter Ayton. 1988. Decision time, subjective probability, and task difficulty. *Memory & Cognition* 16, 2 (1988), 176–185.

[62] Jasy Liew Suet Yan, Nancy McCracken, and Kevin Crowston. 2014. Semi-automatic content analysis of qualitative data. *IConference 2014 Proceedings* (2014).

[63] Ming Yin, Jennifer Wortman Vaughan, and Hanna Wallach. 2019. Understanding the effect of accuracy on trust in machine learning models. In *Proceedings of the 2019 chi conference on human factors in computing systems*. 1–12.

A RESULTS

A.1 Model Performance

Table 6. A mixed two-way ANOVA result for Precision@5 of automatic evaluation of model performance, where * p < 0.05, ** p < 0.01, *** p < 0.001.

Source	SS	DF1	DF2	MS	F	p-unc	np2	eps
code_granularity	0.13	2	27	0.06	2.20	0.13	0.14	
text_granularity	0.15	2	54	0.08	5.80	0.01*	0.18	0.90
Interaction	0.15	4	54	0.04	2.81	0.03*	0.17	

Table 7. Pairwise comparison results for Precision@5 of automatic evaluation of model performance, where * p < 0.05, ** p < 0.01, *** p < 0.001.

code_granularity	A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
-	Sentence	Paragraph	TRUE	TRUE	-1.02	29	two-sided	0.32	0.95	bonf	0.31	-0.2
-	Sentence	Selective	TRUE	TRUE	2.54	29	two-sided	0.02*	0.05*	bonf	2.93	0.52
-	Paragraph	Selective	TRUE	TRUE	2.8	29	two-sided	0.01*	0.03*	bonf	4.91	0.69
Long	Sentence	Paragraph	TRUE	TRUE	-0.54	9	two-sided	0.60	1.00	bonf	0.35	-0.19
Long	Sentence	Selective	TRUE	TRUE	2.80	9	two-sided	0.02	0.19	bonf	3.49	0.93
Long	Paragraph	Selective	TRUE	TRUE	3.24	9	two-sided	0.01	0.09	bonf	6.15	1.15
Mixed	Sentence	Paragraph	TRUE	TRUE	-1.24	9	two-sided	0.25	1.00	bonf	0.57	-0.49
Mixed	Sentence	Selective	TRUE	TRUE	2.58	9	two-sided	0.03	0.27	bonf	2.63	1.30
Mixed	Paragraph	Selective	TRUE	TRUE	2.39	9	two-sided	0.04	0.37	bonf	2.05	1.07
Short	Sentence	Paragraph	TRUE	TRUE	0.10	9	two-sided	0.92	1.00	bonf	0.31	0.04
Short	Sentence	Selective	TRUE	TRUE	-0.76	9	two-sided	0.47	1.00	bonf	0.39	-0.20
Short	Paragraph	Selective	TRUE	TRUE	-0.63	9	two-sided	0.55	1.00	bonf	0.36	-0.24

Table 8. A mixed two-way ANOVA result for Recall@5 of automatic evaluation of model performance, where * p < 0.05, ** p < 0.01, *** p < 0.001.

Source	SS	DF1	DF2	MS	F	p-unc	np2	eps
code_granularity	0.27	2	27	0.13	3.08	0.06	0.19	
text_granularity	0.87	2	54	0.43	19.12	$\leq 0.001^{***}$	0.41	0.86
Interaction	0.10	4	54	0.02	1.06	0.38	0.07	

Table 9. Pairwise comparison results for Recall@5 of human evaluation of model performance, where * p < 0.05, ** p < 0.01, *** p < 0.001.

A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
Sentence	Paragraph	TRUE	TRUE	-2.64	29	two-sided	0.01	0.04	bonf	3.57	-0.64
Sentence	Selective	TRUE	TRUE	3.63	29	two-sided	≤0.001	≤0.001***	bonf	30.71	0.66
Paragraph	Selective	TRUE	TRUE	6.51	29	two-sided	≤0.001	≤0.001***	bonf	41140.00	1.40

Table 10. A mixed two-way ANOVA result for MAP@5 of automatic evaluation of model performance, where * p < 0.05, ** p < 0.01, *** p < 0.001.

Source	SS	DF1	DF2	MS	F	p-unc	p-GG-corr	np2	eps	sphericity	W-spher	p-spher
code_granularity	0.27	2	27	0.14	2.65	0.09		0.16				
text_granularity	1.12	2	54	0.56	20.30	≤0.001	≤0.001***	0.43	0.72	FALSE	0.61	0.0009***
Interaction	0.04	4	54	0.01	0.39	0.81		0.03				

Table 11. Pairwise comparison results for MAP@5 of automatic evaluation of model performance, where * p < 0.05, ** p < 0.01, *** p < 0.001.

A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
Sentence	Paragraph	TRUE	TRUE	-3.38	29	two-sided	≤0.01**	0.01**	bonf	17.22	-0.81
Sentence	Selective	TRUE	TRUE	3.41	29	two-sided	≤0.01**	0.01**	bonf	18.61	0.58
Paragraph	Selective	TRUE	TRUE	6.28	29	two-sided	≤0.001***	≤0.001***	bonf	23300.00	1.34

Table 12. A mixed two-way ANOVA result for Precision@5 of human evaluation of model performance, where * p < 0.05, ** p < 0.01, *** p < 0.001.

Source	SS	DF1	DF2	MS	F	p-unc	np2	eps
code_granularity	0.07	2	27	0.04	1.46	0.25	0.10	
text_granularity	0.01	2	54	0.01	0.21	0.81	0.01	0.86
Interaction	0.06	4	54	0.02	0.51	0.73	0.04	

Table 13. A mixed two-way ANOVA result for MAP@5 of human evaluation of model performance, where * p < 0.05, ** p < 0.01, *** p < 0.001.

Source	SS	DF1	DF2	MS	F	p-unc	p-GG-corr	np2	eps	sphericity	W-spher	p-spher
code_granularity	0.31	2	27	0.16	1.73	0.20		0.11				
text_granularity	1.44	2	54	0.72	37.00	≤0.001	≤0.001***	0.58	0.79	FALSE	0.73	0.01
Interaction	0.22	4	54	0.05	2.82	0.03		0.17				

Table 14. Pairwise comparison results for MAP@5 of human evaluation of model performance, where * p < 0.05, ** p < 0.01, *** p < 0.001.

code_granularity	A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
-	Sentence	Paragraph	TRUE	TRUE	-7	29	two-sided	≤ 0.001	≤ 0.001***	bonf	300000	-1
-	Sentence	Selective	TRUE	TRUE	-4	29	two-sided	≤ 0.001	0.001***	bonf	80	-0.6
-	Paragraph	Selective	TRUE	TRUE	5	29	two-sided	≤ 0.001	≤ 0.001***	bonf	300	0.8
Long	Sentence	Paragraph	TRUE	TRUE	-7	9	two-sided	≤ 0.001	≤ 0.001***	bonf	500	-2
Long	Sentence	Selective	TRUE	TRUE	-3	9	two-sided	0.01	0.1	bonf	5	-0.8
Long	Paragraph	Selective	TRUE	TRUE	3	9	two-sided	0.03	0.2	bonf	3	1
Mixed	Sentence	Paragraph	TRUE	TRUE	-5	9	two-sided	≤ 0.001	0.006**	bonf	60	-2
Mixed	Sentence	Selective	TRUE	TRUE	-2	9	two-sided	0.04	0.4	bonf	2	-0.5
Mixed	Paragraph	Selective	TRUE	TRUE	4	9	two-sided	0.003	0.01**	bonf	3	1
Short	Sentence	Paragraph	TRUE	TRUE	-2	9	two-sided	0.06	0.5	bonf	2	-0.6
Short	Sentence	Selective	TRUE	TRUE	-2	9	two-sided	0.08	0.7	bonf	1	-0.3
Short	Paragraph	Selective	TRUE	TRUE	1	9	two-sided	0.2	1	bonf	0.7	0.4

A.2 Coding Behavior, Decision Time

Table 15. Pairwise comparison results for length of code, where * p < 0.05, ** p < 0.01, *** p < 0.001.

text_granularity	A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
-	Sentence	Paragraph	TRUE	TRUE	-5.64	29	two-sided	<0.001	<0.001***	bonf	4582.827	-0.83
-	Sentence	Selective	TRUE	TRUE	0.47	29	two-sided	0.64	1	bonf	0.215	0.03
-	Sentence	Selective	TRUE	TRUE	0.47	29	two-sided	0.64	1	bonf	0.215	0.03
-	Long	Mixed	FALSE	TRUE	10.12	18	two-sided	<0.001	<0.001***	bonf	9.09E+05	4.33
-	Mixed	Short	FALSE	TRUE	8.22	18	two-sided	<0.001	<0.001***	bonf	5.38E+04	3.52
Sentence	Long	Mixed	FALSE	TRUE	12.61	18	two-sided	<0.001	<0.001***	bonf	2.29E+07	5.4
Sentence	Mixed	Short	FALSE	TRUE	3.35	18	two-sided	0	0.03*	bonf	11.127	1.43
Paragraph	Long	Mixed	FALSE	TRUE	1.41	18	two-sided	0.18	1	bonf	0.788	0.6
Paragraph	Mixed	Short	FALSE	TRUE	12.07	18	two-sided	<0.001	<0.001***	bonf	1.17E+07	5.17
Selective	Long	Mixed	FALSE	TRUE	9.74	18	two-sided	<0.001	<0.001***	bonf	5.34E+05	4.17
Selective	Mixed	Short	FALSE	TRUE	2.48	18	two-sided	0.02	0.21	bonf	2.87	1.06

Table 16. Pairwise comparison results for length of selections, where * p < 0.05, ** p < 0.01, *** p < 0.001.

text_granularity	A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
Selective	Long	Mixed	FALSE	TRUE	5.41	18	two-sided	<0.001	<0.001***	bonf	444.21	2.32
Selective	Long	Short	FALSE	TRUE	3.05	18	two-sided	0.007	0.062	bonf	6.81	1.31
Selective	Mixed	Short	FALSE	TRUE	-0.94	18	two-sided	0.361	1.000	bonf	0.54	-0.40

Table 17. Pairwise comparison results for number of selections, where * p < 0.05, ** p < 0.01, *** p < 0.001.

text_granularity	A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
Selective	Long	Mixed	FALSE	TRUE	-2.563	18	two-sided	0.020	0.117	bonf	3.214	-1.098
Selective	Long	Short	FALSE	TRUE	-2.603	18	two-sided	0.018	0.108	bonf	3.411	-1.115
Selective	Mixed	Short	FALSE	TRUE	-0.223	18	two-sided	0.826	1.000	bonf	0.404	-0.095

Table 18. A mixed two-way ANOVA results for Decision Time, where * p < 0.05, ** p < 0.01, *** p < 0.001.

Source	SS	DF1	DF2	MS	F	p-unc	np2	eps
code_granularity	6675.63	2	24	3337.82	11.13	<0.001***	0.48	
text_granularity	4951.46	2	48	2475.73	10.13	<0.001***	0.30	0.83
Interaction	126.06	4	48	31.52	0.13	0.971	0.01	

Table 19. Pairwise comparison results for Decision Time, where * p < 0.05, ** p < 0.01, *** p < 0.001.

A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
Long	Mixed	FALSE	TRUE	4.297	16	two-sided	0.001	0.002**	bonf	47.53	1.93
Long	Short	FALSE	TRUE	3.276	16	two-sided	0.005	0.014*	bonf	8.97	1.47
Mixed	Short	FALSE	TRUE	-0.555	16	two-sided	0.586	1.000	bonf	0.46	-0.25
Paragraph	Selective	TRUE	TRUE	2.883	26	two-sided	0.008	0.023*	bonf	5.75	0.66
Paragraph	Sentence	TRUE	TRUE	4.208	26	two-sided	0.000	0.001**	bonf	108.38	0.95
Selective	Sentence	TRUE	TRUE	1.635	26	two-sided	0.114	0.342	bonf	0.66	0.33

A.3 Selecting Rate

Table 20. A mixed two-way ANOVA results for Selecting Rate, where * p < 0.05, ** p < 0.01, *** p < 0.001.

Source	SS	DF1	DF2	MS	F	p-unc	np2	eps
code_granularity	0.042	2	27	0.021	0.391	0.680	0.028	
text_granularity	0.403	2	54	0.202	15.838	<0.001***	0.370	0.881
Interaction	0.141	4	54	0.035	2.766	0.036	0.170	

Table 21. Pairwise comparison results for Selecting Rate, where * p < 0.05, ** p < 0.01, *** p < 0.001.

code_granularity	A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
-	Paragraph	Selective	TRUE	TRUE	-4.515	29	two-sided	0.001	<0.001***	bonf	266.048	-0.921
-	Paragraph	Sentence	TRUE	TRUE	-3.507	29	two-sided	0.001	0.004**	bonf	23.215	-0.708
-	Selective	Sentence	TRUE	TRUE	2.186	29	two-sided	0.037	0.111	bonf	1.521	0.337
Long	Paragraph	Selective	TRUE	TRUE	-1.205	9	two-sided	0.259	1	bonf	0.552	-0.327
Long	Paragraph	Sentence	TRUE	TRUE	-0.816	9	two-sided	0.436	1	bonf	0.408	-0.197
Long	Selective	Sentence	TRUE	TRUE	0.832	9	two-sided	0.427	1	bonf	0.412	0.137
Mixed	Paragraph	Selective	TRUE	TRUE	-2.439	9	two-sided	0.037	0.337	bonf	2.197	-0.804
Mixed	Paragraph	Sentence	TRUE	TRUE	-1.417	9	two-sided	0.190	1	bonf	0.674	-0.544
Mixed	Selective	Sentence	TRUE	TRUE	1.140	9	two-sided	0.284	1	bonf	0.521	0.444
Short	Paragraph	Selective	TRUE	TRUE	-4.685	9	two-sided	0.001	0.010**	bonf	36.507	-1.572
Short	Paragraph	Sentence	TRUE	TRUE	-4.653	9	two-sided	0.001	0.011*	bonf	35.185	-1.531
Short	Selective	Sentence	TRUE	TRUE	1.629	9	two-sided	0.138	1	bonf	0.84	0.464

A.4 Perceived Helpfulness

Table 22. A mixed two-way ANOVA results for Perceived Helpfulness, where * p < 0.05, ** p < 0.01, *** p < 0.001.

Source	SS	DF1	DF2	MS	F	p-unc	np2	eps
code_granularity	30.87	2	27	15.43	9.79	0.001**	0.420	
text_granularity	3.20	2	54	1.60	1.53	0.225	0.054	0.941
Interaction	33.13	4	54	8.28	7.94	<0.001***	0.370	

Table 23. Pairwise comparison results for Perceived Helpfulness, where * p < 0.05, ** p < 0.01, *** p < 0.001.

Contrast	code_granularity	A	B	Paired	Parametric	T	dof	alternative	p-unc	p-corr	p-adjust	BF10	hedges
code_granularity	-	Long	Mixed	FALSE	TRUE	4.42	18	two-sided	0.000	0.001**	bonf	72.836	1.892
code_granularity	-	Long	Short	FALSE	TRUE	2.17	18	two-sided	0.044	0.131	bonf	1.855	0.929
code_granularity	-	Mixed	Short	FALSE	TRUE	-2.29	18	two-sided	0.034	0.103	bonf	2.177	-0.980
code_granularity * text_granularity	Long	Paragraph	Selectively	TRUE	TRUE	1.34	9	two-sided	0.213	1.000	bonf	0.626	0.386
code_granularity * text_granularity	Long	Paragraph	Sentence	TRUE	TRUE	0.79	9	two-sided	0.453	1.000	bonf	0.399	0.386
code_granularity * text_granularity	Long	Selectively	Sentence	TRUE	TRUE	0.00	9	two-sided	1.000	1.000	bonf	0.309	0.000
code_granularity * text_granularity	Mixed	Paragraph	Selectively	TRUE	TRUE	6.68	9	two-sided	0.000	0.001**	bonf	310.034	2.376
code_granularity * text_granularity	Mixed	Paragraph	Sentence	TRUE	TRUE	1.71	9	two-sided	0.121	1.000	bonf	0.923	0.798
code_granularity * text_granularity	Mixed	Selectively	Sentence	TRUE	TRUE	-2.75	9	two-sided	0.022	0.202	bonf	3.267	-1.167
code_granularity * text_granularity	Short	Paragraph	Selectively	TRUE	TRUE	-2.94	9	two-sided	0.016	0.148	bonf	4.167	-1.266
code_granularity * text_granularity	Short	Paragraph	Sentence	TRUE	TRUE	-3.07	9	two-sided	0.013	0.119	bonf	4.944	-1.228
code_granularity * text_granularity	Short	Selectively	Sentence	TRUE	TRUE	0.26	9	two-sided	0.798	1.000	bonf	0.318	0.086