Measuring What Matters: Evaluating Ensemble LLMs with Label Refinement in Inductive Coding

Angelina Parfenova[1,2] and **Jürgen Pfeffer**[2]

[1]Lucerne University of Applied Sciences and Arts
[2]Technical University of Munich

Abstract

Inductive coding traditionally relies on labor-intensive human efforts, who are prone to inconsistencies and individual biases. Although large language models (LLMs) offer promising automation capabilities, their standalone use often results in inconsistent outputs, limiting their reliability. In this work, we propose a framework that combines ensemble methods with code refinement methodology to address these challenges. Our approach integrates multiple smaller LLMs, fine-tuned via Low-Rank Adaptation (LoRA), and employs a moderator-based mechanism to simulate human consensus. To address the limitations of metrics like ROUGE and BERTScore, we introduce a composite evaluation metric that combines code conciseness and contextual similarity. The validity of this metric is confirmed through correlation analysis with human expert ratings. Results demonstrate that smaller ensemble models with refined outputs consistently outperform other ensembles, individual models, and even large-scale LLMs like GPT-4. Our evidence suggests that smaller ensemble models significantly outperform larger standalone language models, pointing out the risk of relying solely on a single large model for qualitative analysis.

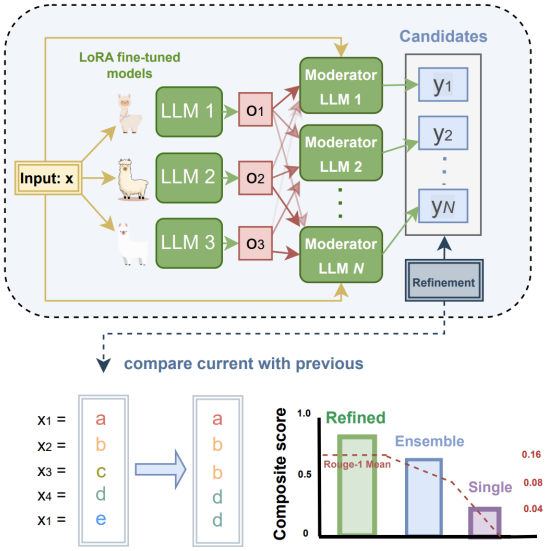

Figure 1: **Overview of the proposed ensemble pipeline for inductive coding.** The input x is processed by multiple LoRA fine-tuned models ($\text{LLM}_1, \text{LLM}_2, \ldots, \text{LLM}_N$), generating candidate outputs ($O_1, O_2, \ldots, O_N$). Moderators compare and evaluate these candidates, selecting $y_1, y_2, \ldots, y_N$ as refined candidates. Refinement of codes consists of comparing the current code with previous ones and assigning the same code based on the chosen similarity threshold.

1 Introduction

Inductive coding is a fundamental method in social science research, enabling the identification, organization, and analysis of patterns and themes within textual data (Creswell, 2016; Saldana, 2016; Braun and Clarke, 2021). At its core, this method extracts key ideas from text, transforming unstructured documents into a list of *codes* that capture the underlying themes. Traditionally, this process has relied on human coders who, despite their expertise, are prone to inconsistencies, cognitive biases, and the tendency to overcomplicate labels (MacQueen and Guest, 2008; Bumbuc and Hrybyk, 2016; Bernard, 2016; Morse, 2017). Large language models (LLMs) offer an opportunity to automate this process, potentially improving efficiency and standardization (Bommasani et al., 2021). However, ensuring the reliability and interpretability of LLM-generated codes remains a challenge.

Our study addresses two critical gaps in automated inductive coding: (1) the need for structured refinement processes to improve consistency across codes and (2) the limitations of standard evaluation metrics, which fail to capture the nuanced criteria used by human experts, such as label granularity, alignment, and contextual relevance (Lin, 2004; Zhang et al., 2019; Sellam et al., 2020; Fabbri et al., 2021a; Parfenova et al., 2024; Mizrahi

Findings of the Association for Computational Linguistics: ACL 2025, pages 10803–10816

et al., 2024). We introduce a novel ensemble-based framework that integrates refinement processes inspired by Retrieval-Augmented Generation (RAG) (Lewis et al., 2020), reusing previously generated codes to eliminate their redundancy iteratively.

This study makes **three key contributions**: (1) we develop a pipeline that integrates multiple instruction-finetuned LLMs with an iterative refinement process to improve coding consistency; (2) we introduce a composite evaluation metric that captures both semantic and structural quality of generated codes; and (3) we demonstrate that smaller LLM ensembles, refined by merging similar codes, outperform standalone large models, achieving closer alignment with human coded benchmarks.

The results indicate that the proposed ensemble approach improves the quality of the generated codes, while the refinement process further enhances their consistency, resulting in outputs that are more closely aligned with the human-coded data. Additionally, we show that ensemble-based methods reduce variability in code assignments, addressing the subjectivity present in individual LLM outputs. These findings provide a strong basis for improving automated qualitative data analysis while maintaining compatibility with human coding standards.

2 Background

Qualitative data analysis (QDA) plays a critical role in social science research (Miller et al., 1990; Creswell, 2016), enabling researchers to identify, categorize and interpret patterns within textual data. Central to this process is the concept of *coding*, where meaningful segments of text are assigned concise labels, or *codes*, that capture their core essence. According to Saldana (2016), a code is "often a word or short phrase that symbolically assigns a summative, salient, essence-capturing and/or evocative attribute to a portion of language-based or visual data." In thematic analysis, one of the most widely used methods in QDA, these codes are further grouped into broader categories to reveal hierarchical relationships and underlying themes within the data (Braun and Clarke, 2021). The source material often includes interviews, documents, or other qualitative data formats, and the process involves summarizing the key ideas in each segment and grouping them into overarching themes.

Recent advances in natural language processing have introduced the use of LLMs for automating qualitative coding tasks (Tornberg, 2023; Parfenova et al., 2024; Fischer and Biemann, 2024). However, we noticed 2 critical points that need to be addressed in this domain. (1) To evaluate codes, some papers noted that traditional summarization metrics like BERT and ROUGE are not good enough for this task (Parfenova et al., 2024; Chen et al., 2024), one research specifically created unsupervised metric to assess codes (Chen et al., 2024). (2) The second critical point in this domain is that although individual LLMs demonstrate remarkable performance, their output often varies due to differences in training data, architectures, and model parameters (Bubeck et al., 2023; Touvron et al., 2023b). This variability introduces subjectivity, making a single LLM's coding decisions as inconsistent as those of individual human annotators. To address these challenges, ensemble methods, techniques that combine multiple models, have been increasingly explored for their ability to take advantage of the complementary strengths of different models and improve performance (Sagi and Rokach, 2018; Jiang et al., 2023b).

Ensemble learning is a widely used strategy to improve model performance by combining the strengths of multiple models, often referred to as "weaker models"(Sagi and Rokach, 2018; Aniol et al., 2019). There are two common approaches to ensemble learning: one involves weighting individual models based on their performance, while the other focuses on aggregating diverse outputs to produce a unified result. One example is the Mix-of-Experts (MoE) framework (Cai et al., 2024), which employs specialized sub-models to make predictions and merges their outputs for improved accuracy. Similarly, LLM-Blender (Jiang et al., 2023a) demonstrates the potential of ensembling by combining ranked outputs from multiple models to achieve superior performance in complex natural language generation tasks.

This study builds upon the concept of ensemble methods but diverges from existing approaches by adopting a *moderator-based framework*. Unlike fusion techniques that combine outputs probabilistically, our approach incorporates a final decision-making model tasked with selecting the best candidate or proposing a novel output. This design reflects the dynamics of human collaboration, where consensus is sometimes driven by a leader or a final arbiter, rather than by averaging or blending

opinions (Engle et al., 2014). Using this moderator model, we aim to mimic the style of the hierarchical decision-making process in human qualitative coding and demonstrate its effectiveness.

3 Dataset

Our experiments used a dataset of 1,000 *code-quote* pairs compiled from two main sources: social science research studies and the SemEval-2014 Task 4 dataset (Pontiki et al., 2014) for augmentation. The social science data includes 600 examples from studies across three universities, featuring topics such as technology interaction, social values, and cultural experiences. The SemEval dataset contributes an additional 400 examples, consisting of reviews manually coded by qualitative researchers.

Table 1 summarizes the data sources. Each *quote* was labeled by 3–5 coders who independently annotated it before reaching a consensus to establish the *golden standard*. This ensured high-quality labels and reduced variability due to individual coder biases. The test set size was set to 100 examples (see Table 2). The dataset was split into training and testing sets without a separate validation set. Hyperparameters were selected based on the training results and evaluated on the test set.

N Quotes	Description
	Social Science Studies Data: 600 quotes
78	Study about interaction with self-tracking devices (interviews)
22	Study about life transitions and mobility (interviews)
82	Study about interaction with voice assistants (interviews)
28	Study about museums and cultural experiences (interviews)
25	Study on doctors' experiences with pregnant women (interviews)
110	Study on universal and national values (interviews)
24	Study on procrastination and budget planning (interviews)
56	Study on technology interactions and user feedback (reviews)
175	Study about social expectations (interviews)
	SemEval 2014; Task 4: 400 quotes
211	Restaurant reviews
189	Laptop reviews

Table 1: Summary of Data Sources with descriptions.

Statistic	Overall	Train	Test
Total Quotes	1000	900	100
Social Science Data	600	550	50
SemEval Data	400	350	50
Num of Data Sources	11	11	11
Unique Codes	680	624	94
Avg. Quote Length	$254.75_{274.28}$	$280.89_{280.89}$	$234.80_{201.61}$
Avg. Code Length	$19.95_{10.43}$	$20.04_{10.70}$	$19.27_{10.53}$

Table 2: Summary statistics of the dataset and train/test splits. Subscript refers to standard deviation where applicable.

To evaluate the ensemble models, we used a second distinct test data set comprising 100 user reviews of ChatGPT sourced from Kaggle (Jikadara, 2023). This data set, characterized by user-generated unstructured feedback, differs from the structured data used for training, providing an opportunity to test model performance on unseen, real-world content. Each review was manually annotated by human coders to create a *golden standard* reference for evaluation. This dataset was chosen specifically for its diversity in style of language, sentiment, and varied quote lengths to verify the ensemble's ability to handle varied text types.

4 Pipeline

Our proposed pipeline consists of two key stages, as illustrated in Figure 1. First, an input x is processed independently by three smaller LLMs (7B and 8B parameter sizes). The outputs from these models are then evaluated by a set of N Moderators ($\text{Moderator}_1, \text{Moderator}_2, \ldots, \text{Moderator}_N$), which refine and consolidate the results. Finally, code merging is performed to ensure consistency across similar inputs, producing the optimal output.

4.1 Phase 1: LoRA Finetuning

In the first phase of the study, we evaluated several open-source models: Llama3 (Touvron et al., 2023a), Falcon (Pineda et al., 2023), Mistral (Team, 2023), Vicuna (Li et al., 2023), Gemma (Team, 2024), and TinyLlama (Jiang et al., 2023c), on an open coding task. We employed various approaches, such as zero-shot, few-shot (providing 1 to 5 examples, see Appendix A), and parameter-efficient fine-tuning (Han et al., 2024) using Low-Rank Adaptation (LoRA) (Hu et al., 2021).

Each model (LLM_i) was fine-tuned on the training dataset and generated an output (o_i) for a given input (x). The outputs were evaluated based on their semantic similarity and token overlap using BERTScore (Zhang et al., 2019) and ROUGE (Lin, 2004). From this evaluation, the three models with the highest performance were selected for the moderation phase (see Appendix B).

4.2 Phase 2: Moderation and Refinement

The top three outputs $\{o_1, o_2, o_3\}$ were passed to Moderators using the prompt template shown in (Appendix C), which incorporated previous model suggestions. The Moderators produced modified outputs $\{y_1, y_2, \ldots, y_N\}$.

To maintain coherence across similar data points, a refinement stage was applied to all generated

Model	Parameters	Adaptation	Prompt	BERTScore			ROUGE		
				P	*R*	*F1*	*1*	*2*	*L*
Llama3 (instruct)	8B	Finetuning	Summarize the main idea of a sentence.	0.72	0.79	0.75	0.18	0.06	0.17
Falcon (instruct)	7B	Finetuning	From the perspective of a social scientist, summarize the following sentence as you would in thematic coding.	0.75	0.79	0.77	0.21	0.09	0.21
Mistral (instruct)	7B	Finetuning	Can you tell me what the main idea of this sentence is in just a few words?	0.74	0.79	0.77	0.25	0.11	0.23
Vicuna (instruct)	7B	Finetuning	Summarize the main idea of a sentence.	0.73	0.78	0.76	0.19	0.07	0.18
Gemma (instruct)	7B	Finetuning	If you were a social scientist doing thematic analysis, what code would you give to this citation?	0.72	0.78	0.75	0.17	0.06	0.17
TinyLlama (chat)	1.1B	Few-shot (5 examples)	Summarize the main idea of a sentence. Here are examples:	0.77	0.74	0.75	0.18	0.03	0.18

Table 3: Performance of various open-source LLMs on open coding task across different adaptation methods and prompts. This table presents the BERTScore and ROUGE scores for each model, indicating precision (*P*), recall (*R*), and F1 scores for BERTScore, along with ROUGE scores (1, 2, L). Models were evaluated under different scenarios, including finetuning and few-shot approaches, with prompts designed to align with thematic analysis. Detailed fine-tuning results are demonstrated in Appendix B.

codes. This process involved code merging, implemented through embedding similarity analysis. Given a new input x, its embedding $\phi(x)$ was computed using sBERT (Reimers and Gurevych, 2019) and compared against previously assigned code embeddings $\phi(p_i)$ using cosine similarity. If the similarity score met or exceeded a predefined threshold ($\text{sim}(\phi(x), \phi(p_i)) \geq \tau$), the existing code was retained; otherwise, a new code was assigned. After generation, each candidate code was scored post hoc using the composite evaluation metric $\mathcal{C}(y_i)$ described in Section 5. This score was used only during benchmarking to compare outputs across methods. It was not used during model training, generation, or selection in real-world applications. For inference without gold labels, the refinement mechanism alone is applied.

5 Metrics

To evaluate the performance of individual models that will serve as input to the moderator, two metrics were employed to capture both lexical and semantic similarity: ROUGE (Lin, 2004) and BERTScore (Zhang et al., 2019). BERTScore is a metric that computes the similarity between BERT token embeddings of two codes, which helps assess the meaning in the generated output compared to the reference. ROUGE is a lexical similarity measure that calculates the overlap of n-grams (1-unigram overlap, 2-bigram overlap, L-longest common subsequence) between the generated text and the reference text. ROUGE is particularly effective in summarization task (Fabbri et al., 2021b), which is valuable when the exact wording of the output needs to match the reference.

5.1 Composite Score

To systematically assess the performance of ensemble models in inductive coding tasks, we introduce a composite score that integrates four evaluation scores: (1) *cosine similarity*, which quantifies the semantic alignment between model-generated codes and human-coded references; (2) *METEOR score*, which captures lexical similarity while accounting for synonyms and stemming variations (Banerjee and Lavie, 2005); (3) *code length penalty*, which discourages excessively verbose codes; and (4) *Jensen-Shannon divergence*, computed between token distributions of reference (golden standard) codes and predicted ones by models. Each component is independently normalized to ensure comparability, and the final composite score is computed as the mean of the four metrics, assigning equal weight to each.

Higher cosine similarity and METEOR scores contribute positively to the composite score, reflecting stronger semantic and lexical agreement with human coders. Conversely, longer codes and greater distributional divergence lower the score, penalizing outputs that deviate from the brevity and consistency typically expected in qualitative coding.

5.2 Metric Validation

To assess the validity of the composite score, we conducted a human evaluation study where experts rated the quality of codes generated by anonymized models (Appendix F). Each expert was presented with ten sentences and their corresponding codes, assigning a rating from 1 to 5 based on clarity, conciseness, and relevance to the text. We then computed Spearman's correlation between the composite score and the average of all expert ratings,

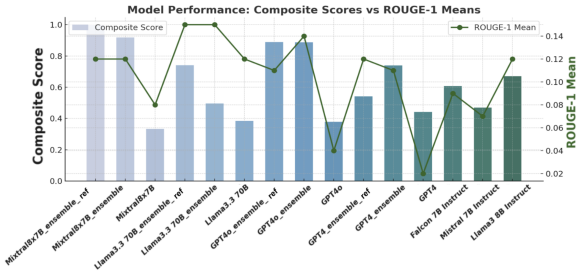

Figure 2: Performance comparison of different models on inductive coding tasks, measured by Composite Scores (bars) and ROUGE-1 Mean (line). Ensemble models with refinement outperform individual models and non-refinement ensembles.

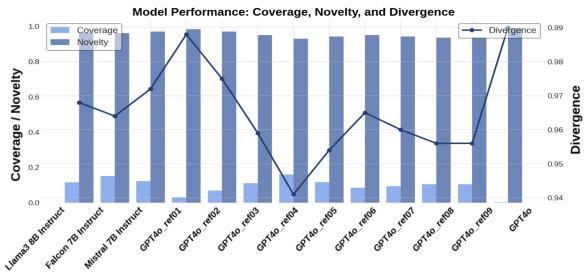

Figure 3: Comparison of LLM performance across three evaluation metrics: *Coverage* (measuring how well a model captures the meaning of the reference codes), *Novelty* (assessing the degree of new information introduced by the model), and Divergence (quantifying distributional shifts from the reference). Ensemble models with code merging tend to balance coverage and novelty better than individual models.

alongside other evaluation metrics.

The analysis reveals a strong correlation between the composite score and human ratings (Spearman: 0.73, p = 0.039), confirming its alignment with expert judgments. ROUGE-1 and ROUGE-L exhibit moderate positive correlations with the composite score (Spearman: 0.51, p = 0.20 and Spearman: 0.44, p = 0.27, respectively), but their statistical insignificance suggests that they may not fully capture the criteria used by human coders. In contrast, BERTScore F1 shows a negligible correlation with the composite score (Spearman: 0.06, p = 0.90), implying that semantic similarity alone is insufficient for assessing inductive coding quality. Additionally, the correlation between the composite score and code length is near zero (Spearman: -0.02, p = 0.96).

Further analysis reveals that BERTScore F1 exhibits a negative correlation with code length, which contradicts the expectation that qualitative codes should be concise (see Table 4). This misalignment suggests that BERTScore may not be an ideal metric for evaluating inductive coding tasks, as it does not adequately penalize redundancy.

6 Results

The ensemble pipeline with codes merging approach yield significant performance improvements, as detailed in Table 6. Ensemble models, particularly refined variants, consistently outperform individual models across all metrics except BERTScore, which remains stable (F1=0.83-0.86) due to its semantic focus. Among standalone models, GPT-4 achieves the highest composite score (0.44), while Llama3.3 70B leads in ROUGE-1 performance (0.12).

Ensembles perform better, with the Mixtral8x7B ensemble achieving a significant composite score improvement (0.33→0.91) over its individual counterpart. Llama3.3 70B shows a similar trend, improving from 0.38 to 0.50 through ensemble integration. Merging of codes provides additional benefits, elevating Mixtral8x7B to 0.99 (+9%) and Llama3.3 70B to 0.74 (+48% from individual baseline). These improvements demonstrate both the value of model diversity and postprocessing effectiveness in enhancing coding consistency.

6.1 Composite Score Analysis

The composite score proved to be effective in capturing semantic, lexical, and structural alignment. Refined ensembles consistently outperformed both individual models and ensembles. Although ROUGE-1 primarily captures lexical similarity, it aligned well with the composite score in the ranking model performance. Also, standalone large models like GPT-4 exhibited lower ROUGE-1 scores compared to ensembles, reflecting challenges in lexical precision.

6.2 Coverage, Novelty, and Divergence Analysis

Our evaluation also incorporates the metrics proposed by Chen et al. (2024), which offer greater sensitivity compared to traditional similarity-based measures like BERTScore (see Figure 3). These metrics reveal nuanced distinctions in model performance, though the observed differences were less sensitive than anticipated, except for Jensen-Shannon divergence—a measure we integrate into our Composite Score. Notably, the results suggest that in certain cases, LLM-generated codes may achieve performance levels comparable to or ex-

Metric	Composite Score	BERTScore F1	ROUGE-1	ROUGE-L	Code Length	Human Rating
Composite Score	1.00	-0.06	0.51	0.44	-0.02	**0.73***
BERTScore F1	-0.06	1.00	-0.09	-0.11	**-0.65***	-0.14
ROUGE-1	0.51	-0.09	1.00	**0.99***	-0.35	0.49
ROUGE-L	0.44	-0.11	**0.99***	1.00	-0.32	0.44
Code Length	-0.02	**-0.65***	-0.35	-0.32	1.00	-0.41
Human Rating	**0.73***	-0.14	0.49	0.44	-0.41	1.00

Table 4: Spearman correlation matrix between evaluation metrics. Statistically significant correlations ($p < 0.1$) are highlighted in **green**. Asterisks (*) indicate significant values.

ceeding human annotations, raising important questions about the reliability of human-labeled benchmarks as definitive gold standards (Chen et al., 2024). However, we refrain from making definitive claims about this observation and instead treat human-generated codes as the established golden standard in our analysis.

6.3 LLMs Alignment

To assess the alignment between different LLMs in qualitative coding, we separately computed the cosine similarity (Figure 5) and Jensen-Shannon divergence (Figure 6) between the codes generated by models. These similarity metrics demonstrate how consistently different models assign qualitative codes to the same textual input and whether ensemble models enhance convergence toward a unified interpretation. The results clearly indicate that ensemble models, particularly GPT-4 Ensemble, Mixtral 8x7B Ensemble, and Llama3.3 70B Ensemble, exhibit higher semantic similarity and lower divergence compared to individual models.

6.4 Qualitative Assessment and Refinement

Table 5 presents a comparative analysis of codes generated by ensemble models against the human-coded gold standard. The GPT-4 Ensemble demonstrates a tendency toward abstract and generalized codes (e.g., "User Satisfaction and App Effectiveness"), which capture broad themes but often lack specificity. In contrast, the Llama3.3 70B and Mixtral 8x7B Ensembles produce codes that align more closely with the gold standard, offering precise labels. This divergence suggests that GPT-4's coding strategy prioritizes thematic abstraction, while smaller ensembles have a more effective balance between abstraction and specificity.

The integration of the postprocessing step into our ensemble framework reduces redundancy in generated codes by aligning new outputs with previously assigned codes. As demonstrated in Table 6, refined ensembles produce more concise outputs,

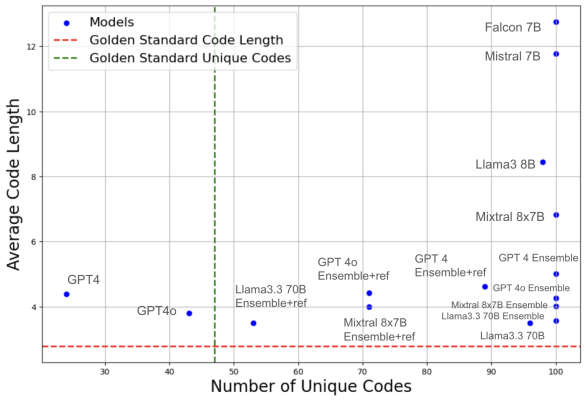

Figure 4: Comparison of LLMs based on the number of unique codes and average code length

achieving an average code length reduction from 6.83 to 4.00 tokens—a 41.5% improvement over standalone models.

Further analysis highlights the impact of post-processing on code diversity. While the human gold standard comprises 47 unique codes with an average length of 2.79 tokens, non-refined models output an excessive number of codes, often generating unique codes for each input. In contrast, code merge integration significantly reduces this redundancy, with Llama3.3 70B Ensemble+ref and Mixtral 8x7B Ensemble+ref producing 53 and 71 unique codes, respectively. This brings the models closer to a human-like coding pattern, as illustrated in Figure 4.

7 Discussion

In this section, we discuss the implications of our findings regarding the application of ensemble LLMs in automated inductive coding. Our results highlight key advantages of ensemble-based approaches over individual models, particularly in terms of conciseness and the ability to balance abstraction and specificity in qualitative coding.

Coder	Golden Standard	Llama3.3 70B Ensemble	GPT4o Ensemble	GPT4 Ensemble	Mixtral 8x7B Ensemble
Quote	**Code Assigned**				
"I am not very happy with this because ChatGPT answers the questions I ask but it stops midway. Then even after continuous efforts it does not give complete answer. This is not very correct. Please fix it so that there is no more trouble. This is a very wrong thing, after some use you are asked to upgrade which is very expensive."	Technical problems while using	Unsatisfied with ChatGPT	User dissatisfaction with incomplete answers	Unsatisfied with ChatGPT's incomplete responses	User: Dissatisfied with expensive, incomplete ChatGPT answers
"This is a great app, the bot is so accurate to anything, it gives me tips in gaming, studies, and life, etc. This app is also helpful if you were having problems in things! Also, there are some problems about the server glitch, but don't worry, some glitches are just seconds, overall a great app. Note: the GPT-4o is more powerful, and accurate than the regular one."	Helpful app	Accurate and helpful app	Helpful and accurate app experience	User Satisfaction and App Efficiency	Accurate, helpful tip-providing app
"The best so far"	Best app	Best so far	The best experience yet	User Satisfaction and App Effectiveness	Top User Rating
"Excellent app for learning"	Excellent for learning	Excellent learning app	Excellent learning tool	User Satisfaction and App Efficiency	Excellent learning app
"Thanks for making my life easier"	Thankfulness	Life made easier	Gratitude for increased ease	User Satisfaction Feedback	Gratitude for Ease

Table 5: Comparison of qualitative codes generated by ensemble models. The golden standard represents human-coded references.

Model	Merge threshold	BERTScore			ROUGE			Composite Score	Code length
		P	*R*	*F1*	*1*	*2*	*L*		
Mixtral8x7B	-	0.83	0.84	0.83	0.08	0.01	0.08	0.33	6.83
Mixtral8x7B Ensemble	-	0.83	0.85	0.84	0.12	0.01	0.08	0.91	4.02
Mixtral8x7B Ensemble + ref	0.7	0.84	0.85	0.84	0.12	0.01	0.11	0.99	4
Llama3.3 70B	-	0.84	0.86	0.85	0.12	0.03	0.12	0.38	3.5
Llama3.3 70B Ensemble	-	0.85	0.86	0.85	0.15	0.02	0.15	0.50	3.57
Llama3.3 70B Ensemble + ref	0.5	0.85	0.88	0.86	0.15	0.03	0.15	0.74	3.49
GPT-4	-	0.83	0.84	0.84	0.02	0.00	0.02	0.44	4.39
GPT-4 Ensemble	-	0.83	0.85	0.84	0.11	0.02	0.10	0.74	5.01
GPT-4 Ensemble + ref	0.8	0.83	0.85	0.84	0.12	0.02	0.10	0.54	4.62
GPT-4o	-	0.85	0.86	0.86	0.04	0.00	0.04	0.37	3.8
GPT-4o Ensemble	-	0.85	0.87	0.86	0.14	0.02	0.14	0.74	4.26
GPT-4o Ensemble + ref	0.7	0.85	0.87	0.86	0.11	0.00	0.11	0.54	4.42
Llama3 8B Instruct	-	0.83	0.86	0.84	0.12	0.02	0.11	0.61	8.45
Falcon 7B Instruct	-	0.83	0.85	0.84	0.09	0.01	0.09	0.47	12.76
Mistral 7B Instruct	-	0.83	0.85	0.84	0.07	0.01	0.07	0.67	11.77

Table 6: Performance comparison of individual models, standard ensembles, and refined ensembles across key metrics. The merge threshold (refinement) column indicates the similarity threshold between the generated code and codes before it (can look only in the past). Models are evaluated using BERTScore (Precision, Recall, and F1), ROUGE (1, 2, and L), Composite Score, and Average Code Length.

7.1 Ensembles Improve Coding Consistency

A major finding of our study is that ensemble models consistently outperform individual models in inductive coding tasks, as shown in Table 6. The improved performance is particularly evident in Jensen-Shannon divergence (Figure 6), where ensembles exhibit lower divergence, indicating a greater degree of alignment. This suggests that aggregating multiple model outputs helps reduce inconsistencies, reflecting the consensus-building process used by human coders in thematic analysis.

The increased consistency observed in ensemble-generated codes aligns with findings from prior research on LLM evaluation, which suggest that individual models often introduce unwanted variability in their outputs due to differences in training data and architectural biases (Bubeck et al., 2023; Jiang et al., 2023b). In contrast, ensemble methods mitigate this variability by integrating diverse inputs, thereby improving robustness. Our results indicate that this effect holds even for smaller models.

7.2 Postprocessing Enhances Code Stability

The integration of code merging significantly improves code stability, as demonstrated by higher composite and ROUGE scores in refined ensembles (Table 6). By referencing previously assigned codes, refinement reduces redundancy for similar inputs. This is particularly evident in the reduction of unique code counts (e.g., 53 for Llama3.3

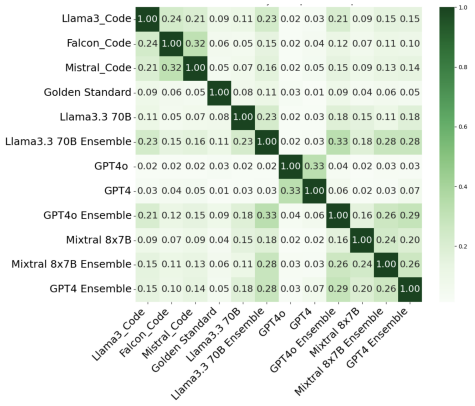

Figure 5: Mean cosine similarity heatmap of model-generated codes. Darker green cells indicate higher semantic similarity between models. Ensemble models (GPT-4 Ensemble, Mixtral 8x7B Ensemble, Llama3.3 70B Ensemble) exhibit the highest similarity, demonstrating greater convergence toward a single interpretation of qualitative codes.

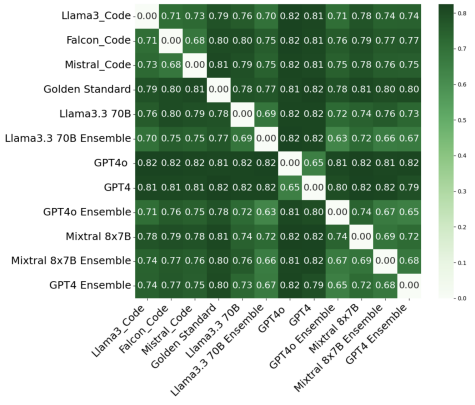

Figure 6: Mean Jensen-Shannon divergence heatmap of model-generated codes. Darker green cells indicate lower divergence, meaning greater agreement in assigned codes. Ensemble models exhibit lower JSD, reinforcing their role in reducing variance and improving convergence in qualitative coding tasks.

70B+ref vs. 100 for non-refined models) and code length (41.5% reduction), bringing model outputs closer to human-like efficiency.

Figure 5 reveals that all ensemble models exhibit the highest similarity scores. This suggests that aggregating inputs from multiple smaller models leads to greater convergence in assigned qualitative codes. These ensembles tend to agree more with each other, reducing variance in code assignments compared to individual models.

7.3 Balancing Abstraction and Specificity in Generated Codes

A key distinction among models is their tendency toward abstraction versus specificity. As shown in Table 5, GPT-4 ensembles often generate broad thematic labels, which, while useful for high-level analysis, may lack the granularity required for domain-specific research. In contrast, smaller ensembles produce more precise codes that closely align with human annotations.

This finding reflects a fundamental trade-off in LLM-based coding: while abstraction improves generalizability, excessive abstraction can obscure critical nuances. Prior work has noted that LLMs trained on diverse corpora tend to favor generalized patterns over domain-specific details (Bubeck et al., 2023; Tornberg, 2023). Results suggest that ensemble approaches can mitigate this issue by combining diverse inputs, thus producing more balanced and contextually grounded outputs.

7.4 Future Directions

Building on our findings, we identify two promising directions for future research. First, hierarchical coding strategies could enable models to generate both broad thematic categories and specific subcategories, better reflecting human coding practices (Saldana, 2016). Second, adaptive RAG mechanisms could dynamically adjust retrieval thresholds based on input complexity, further enhancing consistency and efficiency. These methods could significantly improve the interpretability and quality of automated coding systems.

8 Conclusion

In this work, we explored the use of ensemble-based LLMs with refined outputs for automated inductive coding, demonstrating that ensemble approaches consistently outperform individual models in qualitative coding tasks. Our results highlight three key findings: (1) Ensemble methods improve coding consistency, reducing variance and increasing alignment with human-coded benchmarks; (2) Code merging postprocessing enhances coding stability, ensuring greater coherence in label assignments across similar inputs; and (3) Smaller ensemble models outperform standalone high-capacity models, offering a more efficient and scalable alternative to traditional LLM-based approaches. These findings highlight the potential of ensemble approaches and refinement to advance automated qualitative analysis while maintaining alignment with human coding patterns.

Limitations

While our ensemble-based LLMs demonstrate significant improvements in inductive coding, several limitations warrant consideration.

Bias and Fairness

Although ensembling reduces inconsistencies across individual models, it does not fully address biases inherent in the training data of base models. These biases may manifest in the form of skewed code distributions or overgeneralized labels. Future work should investigate bias mitigation techniques, such as adversarial training or dataset balancing, to further improve model fairness and reliability.

Computational Costs

While our ensemble models are smaller and more efficient than large standalone models like GPT-4, they still require higher computational costs compared to single smaller models. For real-world applications, further optimization strategies—such as dynamic ensemble selection—could help balance accuracy with computational efficiency.

Evaluation Metrics

While our composite metric captures multiple dimensions of coding performance, no single metric fully replicates the nuanced judgment of human coders. Future research should explore more sophisticated evaluation frameworks, incorporating human preference modeling, interactive evaluation setups, or multi-criteria decision analysis to better align automated coding with qualitative research standards.

Metric Accessibility and Fairness

While the composite metric demonstrates strong alignment with human judgments, it requires access to gold-standard codes, which limits its use in real-world deployments. This raises a methodological concern when comparing models that are evaluated using this metric against those that do not rely on gold labels. However, we emphasize that the metric is used only after inference and does not inform code generation or refinement. In future work, we aim to explore unsupervised metrics to enable fully gold-free evaluation.

Language Generalizability

Our study focuses exclusively on English-language datasets, which restricts the generalizability of our findings to other languages and cultural contexts. Extending this approach to multilingual datasets could reveal additional challenges, such as language-specific coding conventions or cultural biases, necessitating further adaptations.

Ethics Statement

The use of LLMs in qualitative research introduces ethical considerations, particularly regarding bias, transparency, and the potential for automated systems to replicate or amplify human biases. While our ensemble-based approach mitigates some of these risks by combining diverse model outputs and employing code refinement for consistency, we acknowledge that biases inherent in training data may still influence results. Human oversight remains critical, as demonstrated by our reliance on expert evaluations to establish a golden standard and validate model outputs. We emphasize that LLMs should serve as assistive tools rather than replacements for human expertise, and we advocate for the development of ethical guidelines to ensure responsible use in sensitive research domains. The datasets used in this study were anonymized and handled in compliance with ethical research standards, minimizing risks to participants.

References

Anna Aniol, Marcin Pietron, and Jerzy Duda. 2019. Ensemble approach for natural language question answering problem. In *2019 Seventh International Symposium on Computing and Networking Workshops (CANDARW)*, pages 180–183. IEEE.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

H. Russell Bernard. 2016. *Research Methods in Anthropology: Qualitative and Quantitative Approaches*. Rowman & Littlefield.

Rishi Bommasani et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Virginia Braun and Victoria Clarke. 2021. *Thematic Analysis: A Practical Guide*. SAGE Publications.

Sebastien Bubeck, Varun Chandak, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Simon Bumbuc and Sam Hrybyk. 2016. Subjectivity in qualitative coding: An issue revisited. *Qualitative Research*, 16(3):253–263.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2024. A survey on mixture of experts.

John Chen, Alexandros Lotsos, Lexie Zhao, Caiyi Wang, Jessica Hullman, Bruce Sherin, Uri Wilensky, and Michael Horn. 2024. A computational method for measuring "open codes" in qualitative analysis.

John W. Creswell. 2016. *30 Essential Skills for the Qualitative Researcher*. SAGE Publications.

Randi A Engle, Jennifer M Langer-Osuna, and Maxine McKinney de Royston. 2014. Toward a model of influence in persuasive discussions: Negotiating quality, authority, privilege, and access within a student-led argument. *Journal of the Learning Sciences*, 23(2):245–268.

Alexander Fabbri et al. 2021a. Summac: Re-visiting nli-based models for consistency evaluation. *Proceedings of EMNLP 2021*.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021b. Summeval: Re-evaluating summarization evaluation.

Tim Fischer and Chris Biemann. 2024. Exploring large language models for qualitative data analysis. In *Proceedings of the 4th International Conference on Natural Language Processing for Digital Humanities*, pages 423–437, Miami, USA. Association for Computational Linguistics.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023a. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion.

Dongfu Jiang et al. 2023b. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. *Proceedings of ACL 2023*.

Zhe Jiang et al. 2023c. Tinyllama: Distilling large language models for efficiency. *arXiv preprint arXiv:2310.05637*.

Bhavik Jikadara. 2023. Chatgpt user feedback dataset. `https://www.kaggle.com/datasets/bhavikjikadara/chatgpt-user-feedback`. Dataset containing user reviews and ratings for the ChatGPT Android App, updated daily. Includes attributes such as review text, numerical ratings, thumbs up counts, and timestamps. Licensed under MIT.

Patrick Lewis et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Proceedings of NeurIPS 2020*.

Chenghao Li, Fangchen Xie, et al. 2023. Vicuna: An open-source chatbot. *FastChat: Open Assistant*. Available at https://vicuna.lmsys.org/.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries.

Kathleen M. MacQueen and Greg Guest. 2008. *Handbook for Team-Based Qualitative Research*. AltaMira Press.

George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. 1990. Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? a call for multi-prompt LLM evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949.

Janice M. Morse. 2017. Considering theory derived from qualitative research. *Qualitative Health Research*, 27(10):1372–1380.

Diana Parfenova et al. 2024. Automating qualitative analysis with llms. *Proceedings of ACL 2024*.

Felipe Pineda, Raphael Milliere, Andreas Vlachos, Myle Ott, Richard Yates, Amelia Glaese, et al. 2023. The falcon series of language models. *arXiv preprint arXiv:2306.01116*.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Omer Sagi and Lior Rokach. 2018. Ensemble learning: A survey. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 8(4):e1249.

Johnny Saldana. 2016. *The Coding Manual for Qualitative Researchers*. SAGE Publications.

Thibault Sellam et al. 2020. Bleurt: Learning robust metrics for text generation. *Proceedings of ACL 2020*.

Gemma AI Research Team. 2024. Gemma: An instructable, open-source large language model.

Mistral AI Team. 2023. Mistral: Efficient pretraining of transformer language models.

Petter Tornberg. 2023. Using large language models for automated qualitative coding in the social sciences. *Nature Machine Intelligence*, 5:576–586.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Thibaut Lavril, et al. 2023b. Llama: Open and efficient foundation language models. In *Proceedings of the 2023 Annual Conference on Machine Learning (ICML)*, pages 123–134.

Tianyi Zhang et al. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Prompts used for finetuning

In the conducted experiments, several prompts were designed to guide the models in performing open coding tasks. These prompts varied in their level of explicitness, the perspective they asked the model to adopt, and the amount of detail they requested. Additionally, the experiments compared the effect of using a line break versus a period (dot) at the end of each prompt to assess how subtle changes in prompt formatting might influence the model's performance (see Appendix B). Below is a brief description of each prompt:

- **Explicit Instruction (Prompt 1):** *Summarize the main idea of a sentence.* This prompt provides a direct and clear instruction to the model, asking it to summarize the core idea of a given sentence. The expectation is for the model to extract the primary message or theme conveyed in the sentence with no additional context or framing. This prompt is designed to test the model's ability to perform a straightforward task without needing implicit knowledge.

- **Informal Request (Prompt 2):** *Can you tell me what the main idea of this sentence is in just a few words?* This prompt is phrased as a casual, conversational question, asking the model to summarize the sentence in "just a few words." The informal tone encourages a more concise and simplified response, aiming to capture how well the model can extract the essence of the sentence in a more natural, everyday context.

- **Expert Angle (Prompt 3):** *From the perspective of a social scientist, summarize the following sentence as you would in thematic coding.* This prompt takes a more specialized approach, asking the model to assume the perspective of a social scientist performing thematic coding. The expectation here is for the model to not only summarize the sentence but to apply a more analytical and structured lens, possibly introducing higher-level categorizations that would be typical in qualitative data analysis.

- **Impersonalization (Prompt 4):** *If you were a social scientist doing thematic analysis, what code would you give to this citation?* In this prompt, the model is asked to act as a social scientist and assign a code, which is a brief label representing the central idea of the sentence. It emphasizes the objectivity of thematic analysis, expecting the model to depersonalize the task and focus on generating an appropriate label that accurately reflects the content.

- **Detailed Explanation (Prompt 5):** *Explain in a couple of words the primary thought expressed in the following text.* This prompt asks the model to provide a more detailed, thorough explanation of the primary thought behind the text. It is designed to encourage the model to go beyond a simple summary and delve into the deeper meaning or implications of the sentence.

- **Simplified Task (Prompt 6):** *What is the gist of this sentence?* This prompt simplifies the task by asking for the "gist" of the sentence. It challenges the model to provide a very brief and straightforward summary, focusing on distilling the essential meaning of the sentence.

B Detailed fine-tuning results

These results (see Table 7) demonstrate the performance of various models when fine-tuned on the task of open coding using different prompts. BERTScore and ROUGE are reported.

C Moderator prompt template

D Links to models on Hugging Face

- **Llama3 8B**: https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

Model	BERTScore			ROUGE		
	P_{std}	R_{std}	$F1_{std}$	1	2	L
Summarize the main idea of a sentence\n						
Llama3	$0.713_{0.060}$	$0.758_{0.040}$	$0.734_{0.062}$	0.141	0.033	0.153
Falcon	$0.746_{0.073}$	$0.782_{0.097}$	$0.764_{0.095}$	0.176	0.047	0.189
Mistral	$0.729_{0.076}$	$0.787_{0.093}$	$0.756_{0.078}$	0.178	0.047	0.195
Vicuna	$0.731_{0.063}$	$0.777_{0.095}$	$0.753_{0.079}$	0.163	0.028	0.182
Gemma	$0.712_{0.084}$	$0.738_{0.078}$	$0.745_{0.080}$	0.163	0.030	0.168
TinyLlama	$0.718_{0.072}$	$0.775_{0.090}$	$0.757_{0.087}$	0.164	0.052	0.158
Summarize the main idea of a sentence.						
Llama3	$0.718_{0.072}$	$0.788_{0.089}$	$0.750_{0.073}$	0.181	0.059	0.166
Falcon	$0.738_{0.099}$	$0.787_{0.103}$	$0.761_{0.096}$	0.213	0.077	0.210
Mistral	$0.719_{0.072}$	$0.768_{0.086}$	$0.742_{0.075}$	0.157	0.055	0.148
Vicuna	$0.733_{0.079}$	$0.787_{0.095}$	$0.758_{0.081}$	0.193	0.068	0.185
Gemma	$0.719_{0.071}$	$0.779_{0.089}$	$0.746_{0.072}$	0.172	0.049	0.166
TinyLlama	$0.736_{0.083}$	$0.788_{0.092}$	$0.760_{0.081}$	0.207	0.074	0.199
Can you tell me what the main idea of this sentence is in just a few words?						
Llama3	$0.688_{0.055}$	$0.778_{0.084}$	$0.729_{0.061}$	0.116	0.034	0.110
Falcon	$0.753_{0.105}$	$0.787_{0.108}$	$0.768_{0.102}$	0.236	0.104	0.239
Mistral	$0.742_{0.106}$	$0.795_{0.106}$	$0.766_{0.101}$	0.246	0.106	0.235
Vicuna	$0.691_{0.060}$	$0.783_{0.087}$	$0.732_{0.063}$	0.168	0.047	0.164
Gemma	$0.711_{0.075}$	$0.786_{0.093}$	$0.746_{0.079}$	0.171	0.057	0.168
TinyLlama	$0.725_{0.083}$	$0.789_{0.090}$	$0.754_{0.079}$	0.178	0.067	0.177
From the perspective of a social scientist, summarize the following sentence as you would in thematic coding\n						
Llama3	$0.698_{0.059}$	$0.784_{0.083}$	$0.738_{0.062}$	0.130	0.033	0.119
Falcon	$0.745_{0.109}$	$0.792_{0.105}$	$0.766_{0.102}$	0.210	0.089	0.211
Mistral	$0.688_{0.060}$	$0.785_{0.086}$	$0.732_{0.064}$	0.139	0.041	0.131
Vicuna	$0.713_{0.080}$	$0.778_{0.094}$	$0.743_{0.080}$	0.169	0.061	0.166
Gemma	$0.721_{0.085}$	$0.784_{0.093}$	$0.749_{0.082}$	0.180	0.070	0.177
Tinyllama	$0.718_{0.073}$	$0.776_{0.083}$	$0.745_{0.072}$	0.165	0.053	0.158
From the perspective of a social scientist, summarize the following sentence as you would in thematic coding.						
Llama3	$0.685_{0.082}$	$0.781_{0.064}$	$0.733_{0.081}$	0.136	0.025	0.154
Falcon	$0.754_{0.066}$	$0.778_{0.091}$	$0.759_{0.088}$	0.181	0.048	0.190
Mistral	$0.740_{0.067}$	$0.780_{0.088}$	$0.756_{0.071}$	0.172	0.045	0.187
Vicuna	$0.718_{0.071}$	$0.780_{0.094}$	$0.753_{0.073}$	0.165	0.039	0.185
Gemma	$0.700_{0.072}$	$0.780_{0.085}$	$0.746_{0.080}$	0.180	0.046	0.187
TinyLlama	$0.729_{0.076}$	$0.778_{0.089}$	$0.754_{0.080}$	0.169	0.046	0.183
If you were a social scientist doing thematic analysis, what code would you give to this citation?						
Llama3	$0.692_{0.060}$	$0.785_{0.083}$	$0.735_{0.064}$	0.064	0.043	0.126
Falcon	$0.736_{0.093}$	$0.785_{0.101}$	$0.759_{0.092}$	0.206	0.076	0.200
Mistral	$0.686_{0.057}$	$0.785_{0.082}$	$0.731_{0.061}$	0.132	0.044	0.123
Vicuna	$0.719_{0.070}$	$0.789_{0.091}$	$0.751_{0.073}$	0.183	0.063	0.169
Gemma	$0.724_{0.085}$	$0.784_{0.091}$	$0.751_{0.082}$	0.170	0.066	0.168
Tinyllama	$0.720_{0.071}$	$0.778_{0.083}$	$0.747_{0.072}$	0.186	0.053	0.182
What is the gist of this sentence?						
Llama3	$0.680_{0.064}$	$0.780_{0.086}$	$0.725_{0.066}$	0.129	0.042	0.121
Falcon	$0.731_{0.091}$	$0.780_{0.098}$	$0.754_{0.089}$	0.182	0.080	0.179
Mistral	$0.726_{0.079}$	$0.785_{0.095}$	$0.753_{0.079}$	0.165	0.057	0.160
Vicuna	$0.720_{0.070}$	$0.781_{0.089}$	$0.748_{0.072}$	0.172	0.055	0.162
Gemma	$0.707_{0.077}$	$0.773_{0.091}$	$0.737_{0.076}$	0.152	0.059	0.146
Tinyllama	$0.713_{0.057}$	$0.773_{0.079}$	$0.741_{0.061}$	0.143	0.032	0.139
Explain in a couple of words the primary thought expressed in the following text\n						
Llama3	$0.691_{0.062}$	$0.783_{0.085}$	$0.733_{0.066}$	0.120	0.038	0.110
Falcon	$0.734_{0.078}$	$0.778_{0.090}$	$0.754_{0.078}$	0.171	0.049	0.165
Mistral	$0.698_{0.067}$	$0.780_{0.088}$	$0.735_{0.070}$	0.141	0.038	0.131
Vicuna	$0.703_{0.072}$	$0.780_{0.088}$	$0.738_{0.072}$	0.155	0.048	0.148
Gemma	$0.706_{0.064}$	$0.786_{0.086}$	$0.742_{0.066}$	0.177	0.053	0.170
Tinyllama	$0.720_{0.077}$	$0.784_{0.091}$	$0.750_{0.078}$	0.168	0.071	0.163
Explain in a couple of words the primary thought expressed in the following text.						
Llama3	$0.700_{0.068}$	$0.784_{0.055}$	$0.747_{0.063}$	0.142	0.025	0.152
Falcon	$0.752_{0.088}$	$0.779_{0.061}$	$0.760_{0.086}$	0.183	0.042	0.193
Mistral	$0.738_{0.070}$	$0.790_{0.090}$	$0.759_{0.073}$	0.173	0.047	0.183
Vicuna	$0.717_{0.066}$	$0.780_{0.094}$	$0.752_{0.099}$	0.161	0.025	0.182
Gemma	$0.708_{0.068}$	$0.778_{0.079}$	$0.746_{0.098}$	0.172	0.039	0.186
TinyLlama	$0.728_{0.073}$	$0.778_{0.091}$	$0.755_{0.089}$	0.168	0.053	0.168

Table 7: Detailed Fine-tuning Results. The following table presents the detailed results from fine-tuning experiments, including precision (P), recall (R), F1 score, and ROUGE across different models and prompts.

```
You will be given a paragraph from the
    text, which is: {textdescription}.

Definition of the code: A word or short
    phrase that symbolically
assigns a summative, salient, essence-
    capturing, and/or evocative
attribute for a portion of language-
    based or visual data.

Here is the excerpt to code:
{row['Paragraph']}

Here are three coding suggestions from
    previous models:
1. {row['Llama3_Code']}
2. {row['Falcon_Code']}
3. {row['Mistral_Code']}

Please choose the best code or suggest a
     new code taking into account all
    these answers.
The output should be the code with no
    longer than 5 words.
```

Listing 1: Moderator Prompt Template with Model Suggestions

- **Falcon 7B**: https://huggingface.co/tiiuae/falcon-7b-instruct
- **Mistral 7B**: https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2
- **Vicuna 7B**: https://huggingface.co/lmsys/vicuna-7b-v1.5
- **Gemma 7B**: https://huggingface.co/google/gemma-7b-it
- **TinyLlama 1.1B**: https://huggingface.co/TinyLlama/TinyLlama-1.1B-Chat-v1.0
- **Llama 3.3. 70B**: https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct
- **Mixtral 7x8B**: https://huggingface.co/mistralai/Mixtral-8x7B-v0.1

E LoRA configuration

This section provides the LoRA (Low-Rank Adaptation) configuration used for fine-tuning the models in this study. Below is the code snippet used for configuring LoRA:

```
config = LoraConfig(
    r=16,
    lora_alpha=32,
    target_modules=["gate_proj", "up_proj", "down_proj"],
    lora_dropout=0.05,
    bias="none",
    task_type="CAUSAL_LM"
)

model = get_peft_model(model, config)
print_trainable_parameters(model)

generation_config = model.generation_config
generation_config.max_new_tokens = 15
generation_config.temperature = 0.7
generation_config.top_p = 0.7
generation_config.num_return_sequences = 1
generation_config.pad_token_id = tokenizer.eos_token_id
generation_config.eos_token_id = tokenizer.eos_token_id
```

F Human evaluation instructions

To validate the quality of automatically generated qualitative codes, we conducted a human evaluation study in which domain experts rated model-generated codes for a given set of quotes. The evaluation form was structured as follows:

F.1 Code Rating Guidelines

Introduction:

Thank you for participating in this evaluation study. Your expertise is crucial in assessing the quality of automatically generated qualitative codes. In this task, you will be presented with a sentence (quote) and a corresponding code assigned by a model. You will rate the quality of the code based on its accuracy, relevance, and conciseness.

Definition of a Code:

According to Saldana (2016), a code is "often a word or short phrase that symbolically assigns a summative, salient, essence-capturing, and/or evocative attribute for a portion of language-based or visual data."

F.2 Instructions for Participants

1. **Read the given quote carefully.**
2. **Review the assigned code.**
3. **Rate the quality of the code** on a scale from 1 to 5 based on the following criteria:

- **1 - Very Poor:** The code does not reflect the main idea of the quote at all.
- **2 - Poor:** The code partially captures the quote but lacks clarity or relevance.
- **3 - Acceptable:** The code is somewhat relevant but could be improved for clarity or conciseness.
- **4 - Good:** The code accurately represents the key idea of the quote with minimal ambiguity.
- **5 - Excellent:** The code perfectly captures the essence of the quote in a concise and meaningful way.

These ratings were collected for all models across a diverse set of ten quotes. The results were analyzed to compute correlations between human ratings and various automated evaluation metrics, contributing to the validation of the composite score.