Generative AI for Peer Assessment Helpfulness Evaluation

Chengyuan Liu
North Carolina State University
cliu32@ncsu.edu

Jialin Cui
North Carolina State University
jcui9@ncsu.edu

Ruixuan Shang
North Carolina State University
rshang@ncsu.edu

Qinjin Jia
North Carolina State University
qjia3@ncsu.edu

Parvez Rashid
North Carolina State University
mrashid4@ncsu.edu

Edward Gehringer
North Carolina State University
efg@ncsu.edu

ABSTRACT

Evaluating the helpfulness of review comments is increasingly important in peer-assessment research, as students are more likely to accept and implement the feedback they perceive as helpful. Automating the evaluation of review helpfulness by AI models faces two challenges: (1) the limited availability of annotated datasets with helpfulness tags for supervised model training; (2) the subjective, often ambiguous nature of helpfulness, which complicates achieving consistency in evaluations. Although previous studies have investigated related review-comment features and their correlation with perceived helpfulness, an explicit evaluation method is still lacking. Our study employs generative AI, specifically OpenAI's GPT models, to evaluate the review helpfulness through text generation based on input prompts. We also apply prompt engineering to guide the model toward producing more relevant and desired responses through well-defined prompt templates, which can significantly reduce the need for training datasets. We investigate Chain-of-Thought (CoT) and few-shot prompting (strategies of prompt engineering) to optimize the model output, by incorporating a problem-solving reasoning process and concrete examples for demonstration. Additionally, we propose "**R**easoning **a**fter **F**eature **I**dentification" (**RaFI**) framework, an innovative multi-step reasoning process designed to optimize the CoT prompting, by identifying key comment features then applying logical reasoning. Our results show that RaFI outperforms traditional supervised methods and basic CoT prompting in evaluating review helpfulness, requiring no model training and generating better responses, showcasing its potential as a transformative tool in AI-driven peer-assessment analysis.

Keywords

generative AI, peer assessment, helpfulness, prompt engineering, reasoning

C. Liu, J. Cui, R. Shang, Q. Jia, P. Rashid, and E. Gehringer. Generative ai for peer assessment helpfulness evaluation. In B. Paaßen and C. D. Epp, editors, *Proceedings of the 17th International Conference on Educational Data Mining*, pages 412–419, Atlanta, Georgia, USA, July 2024. International Educational Data Mining Society.

1. INTRODUCTION

Peer assessment, widely utilized in various educational scenarios, has proven highly effective in improving student learning outcomes and facilitating instructors' grading processes [15]. In this system, students are asked to provide both numerical ratings and textual review comments on their peers' work. These textual review comments with their helpful and constructive features, have gained significant attention in peer-assessment research due to their potential to encourage students to think critically about their work and further improve [8]. Studies have demonstrated the importance of automatically evaluating the helpfulness of these review comments, highlighting that students are more likely to implement the feedback and refine their work when they perceive the comments as helpful [24]. However, automatically evaluating helpfulness poses challenges due to its inherently ambiguous and subjective nature [25]. Training language models to simulate human evaluators and understand the underlying logic in helpfulness evaluation is an intricate undertaking. Furthermore, collecting a labeled dataset with helpfulness tags for supervised model training can be challenging as well [37].

To address those challenges, researchers have tried to implicitly identify related features of review comments. Examples of such features are "mentions problem" or "offers suggestion"; these features have been correlated with perceived helpfulness of review comments containing them [15, 39]. However, no prior research has focused on integrating these identified features with clear logic to explicitly evaluate the review's helpfulness. This gap exists because supervised language models are typically limited to specific tasks for which they are trained on and lack the capability for information integration and language comprehension.

Generative AI models, typically trained on a vast and diverse range of text resources, are proficient in understanding and handling various language tasks through text generation [3]. Their universal language perception makes them more flexible and adaptable to different language tasks without the need for task-specific training and labeled datasets, which are commonly challenging to collect. Prompt engineering is a strategy that optimizes the AI model's response by providing a well defined input text (as known as prompt) [28].

This approach enables us to compile the input data along with decision-making logic (i.e., evaluating review helpfulness) into the prompt, guiding the generative AI model to understand the task and generate the desired response. We assessed the effectiveness of two prompt engineering strategies: Chain-of-Thought (CoT) prompting [34], which elicits the model's reasoning capability to mimic the problem-solving process of human evaluators, and few-shot prompting [2], which uses a small number of examples to demonstrate the task and reinforce understanding.

However, the traditional single-step reasoning process, incorporating problem-solving steps into a narrative prompt, does not adequately address all types of review comments and their features. Thus, we propose a novel framework, "**R**easoning **a**fter **F**eature **I**dentification" (**RaFI**), as an advanced two-step reasoning process for evaluating review helpfulness. This framework involves first identifying features in review comments based on existing research, then logically integrating these features through reasoning to determine helpfulness. Our results demonstrate that the RaFI framework achieves a 22.98% increase in accuracy and a 27.79% improvement in $f1$ score on average, compared to traditional CoT prompting across various scenarios, including zero-shot, one-shot, and few-shot settings.

2. RELATED WORK

2.1 Peer Assessment Helpfulness Evaluation

Peer assessment has proven to be an effective tool in enhancing student learning, as it provides high-quality, constructive feedback that helps students recognize the strengths and weaknesses in their work, thus improving their knowledge and output [40, 8, 13, 7]. The quality of peer-assessment comments has been measured by identifying key characteristics in review comments, such as summarization, problem identification, solution suggestion, localization, explanation, scope, praise, and mitigating language [22]. While natural language processing and deep learning have been used for automating this analysis, there is a research gap in evaluating the helpfulness of review comments due to their subjective nature. Studies have investigated the correlation between feedback helpfulness and those characteristics in review comments, suggesting that characteristics like problem localization and suggestions are particularly useful for student improvement [22]. Patchan et al. [25] and Xiao et al. [37] both found a strong relationship between helpful reviews and features like problem statements, localization, suggestions, and praise, indicating that the absence of these elements often leads to unhelpful feedback.

2.2 Large Language Models for Text Classification

Large Language Models (LLMs) are advanced AI systems employing deep learning techniques and trained with extensive text datasets, proficient in understanding and generating human-like text. The success of LLMs is attributed to the transformer architecture [32], which effectively transforms the input sequences into outputs (i.e., machine translation) while capturing word relations for context-aware language understanding. Models like BERT (Bidirectional Encoder Representations from Transformers) [6], an encoder-only transformer model, focus on context understanding and word relationships through Masked Language Modeling and Next Sentence Prediction. In contrast, the decoder-only GPT models (Generative Pre-trained Transformer)
like GPT-3 [27], demonstrate superior text-generation capabilities, aided by their large-scale pre-training. Text classification [20, 11]is one of the most common tasks in NLP (e.g., sentiment analysis). Fine-tuning BERT models to understand the context of domain-specific text data has proven highly effective in text classification tasks [31, 3]. However, these tasks can be challenging when there is insufficient labeled data available for the supervised model fine-tuning [35, 16]. GPT-3 has been trained on 45 TB of text data from multiple sources, which include Wikipedia and books, comprising 175 billion parameters [2]. This enables a significant advancement in many language tasks including text classification. Its comprehensive language understanding and text generation abilities allow for more efficient adaptation to new contexts with less reliance on labeled data [18].

2.3 Prompt Engineering and In-context Learning

Prompt engineering utilizes prompts (input messages to generative AI model) in certain formats to tailor generative AI models for particular tasks, ensuring the production of accurate and relevant responses [36, 14, 26]. This method eliminates the necessity of optimizing the model's parameters. Zero-shot prompting [10] relies exclusively on the model's pre-existing knowledge to generate responses, providing only the task description in the prompt. This method has shown effectiveness in numerous language tasks [21, 30], given that the large language model (LLM) has been pre-trained on a large text corpus. In contrast, few-shot prompting [28] provides more precise guidance by including a few examples. This is also known as in-context learning [19]. This approach helps the model to learn the input-output mapping through the given examples. Incorporating explanations into the few-shot examples [12] has been shown to enhance the model's comprehension of the context and the logic behind these examples. CoT prompting [34, 33] further refines this approach by encouraging the model to generate intermediate steps and reasoning, thereby producing responses that are both more accurate and interpretable.

3. METHODOLOGY

3.1 Prompt Template

3.1.1 Overview

Our study focuses on a binary text-classification task, which requires a carefully designed prompt template. This template must incorporate some key elements that guide the generative AI model to accurately categorize the given text or demonstrate the desired output. These elements include:

(1) **Task Description.** To guide the language model to understand and perform the task effectively, it is essential to start with a clear definition and task description, denoted as x_{desc}. This facilitates a concise way of indicating what kind of classification is being requested.

(2) **Input.** Unlike traditional text-classification tasks, which simply assign an output label y_{label} to a given piece of input text x_{input} (e.g., sentiment analysis), evaluating the helpfulness of review comments requires consideration of whether

the comments follow certain guidelines. The rubric items x_{rubric} specify criteria that the review comments need to follow. This is crucial because a comment that is helpful in one scenario (e.g., in relation to a specific rubric item) may not be helpful in another, as demonstrated in the examples below:

> Rubric #1: "Please comment on how the code follows the DRY development principle."
> Rubric #2: "Please comment on one of the good parts of the code."
> Review: "The code follows the DRY development principle."
> (Not helpful for rubric #1 but helpful for rubric #2)

(3) **Context.** A well-designed prompt should also include examples as context to demonstrate the desired output format. These examples concretely show how to correctly categorize a given text. It is beneficial to include examples from both categories (helpful and not helpful). The format for these examples is as follows:

$$\left\{(x^0_{input}, x^0_{rubric}, y^0_{label}), ..., (x^i_{input}, x^i_{rubric}, y^i_{label})\right\} \quad (1)$$

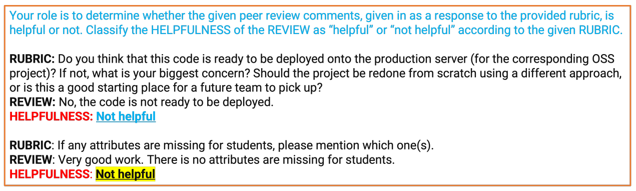
where x^i_{input} and x^i_{rubric} $(0 < i < j)$ denotes the provided review comments and the corresponding rubric. j indicates the number of annotated examples provided, while y^i_{label} denotes the ground-truth helpfulness label (assigned as either 0 or 1) of the review comments.

3.1.2 Zero-shot scenario

In this study, we evaluate multiple prompt templates with zero-shot, one-shot, and few-shot settings. In the zero-shot setting, the language model is only given the task description x_{desc} along with the input comment x_{input} and rubric x_{rubric}, to predict the helpfulness label y_{label}.

3.1.3 One-shot/few-shot scenario

In comparison to the zero-shot setting, the one-shot and few-shot settings additionally introduce a small number of annotated examples as context $(x^i_{input}, x^i_{rubric}, y^i_{label})$ $(0 < i < j)$ for the language model. In the one-shot setting, only one example is provided ($j = 1$). In the few-shot setting, we empirically use $j = 4$ as an optimal number of examples. The prompt template and a one-shot example is illustrated in Figure 1.



Your role is to determine whether the given peer review comments, given in as a response to the provided rubric, is helpful or not. Classify the HELPFULNESS of the REVIEW as "helpful" or "not helpful" according to the given RUBRIC.

RUBRIC: Do you think that this code is ready to be deployed onto the production server (for the corresponding OSS project)? If not, what is your biggest concern? Should the project be redone from scratch using a different approach, or is this a good starting place for a future team to pick up?
REVIEW: No, the code is not ready to be deployed.
HELPFULNESS: **Not helpful**

RUBRIC: If any attributes are missing for students, please mention which one(s).
REVIEW: Very good work. There is no attributes are missing for students.
HELPFULNESS: **Not helpful**

Figure 1: Vanilla one-shot example. The designed prompt message and the provided prediction in the example are indicated in blue text, whereas the response generated by the model is in yellow.

3.2 Chain-of-Thought (CoT) prompting

3.2.1 Overview

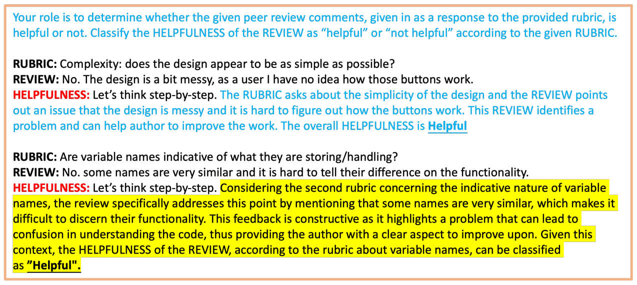
For tasks like evaluating review helpfulness, jumping directly to an answer (helpful or not helpful) can lead to errors. Chain-of-thought (CoT) prompting is a technique that guides language models to provide additional reasoning steps when tackling complex tasks [34]. The cornerstone of implementing CoT prompting is to provide clear, step-by-step reasoning mirroring human problem-solving strategies.

3.2.2 Zero-shot scenario

Zero-shot is a relatively unconventional setting in Chain-of-Thought (CoT) prompting, as it is formulated without providing any annotated examples or explicit reasoning. This challenge can be mitigated by incorporating the phrase **"Let's think step by step"** into the prompt [10]. This addition can also encourage the model to think about the rationale behind a decision even without examples provided.

3.2.3 One-shot/few-shot scenario

In the one-shot/few-shot setting, the reasoning process accompanies the examples provided. However, this step is not originally present in the data, as when students annotate the helpfulness of the review comments they received, they typically provide only a binary label indicating the helpfulness, without offering the underlying logic that led to their conclusion. To develop this reasoning process, we can leverage insights from existing research that delves into the features (characteristics) of what makes a review comment helpful [22, 25, 23]. The example of one-shot Chain of Thought (CoT) prompting is illustrated in Figure 2.



Your role is to determine whether the given peer review comments, given in as a response to the provided rubric, is helpful or not. Classify the HELPFULNESS of the REVIEW as "helpful" or "not helpful" according to the given RUBRIC.

RUBRIC: Complexity: does the design appear to be as simple as possible?
REVIEW: No. The design is a bit messy, as a user I have no idea how those buttons work.
HELPFULNESS: Let's think step-by-step. The RUBRIC asks about the simplicity of the design and the REVIEW points out an issue that the design is messy and it is hard to figure out how the buttons work. This REVIEW identifies a problem and can help author to improve the work. The overall HELPFULNESS is **Helpful**

RUBRIC: Are variable names indicative of what they are storing/handling?
REVIEW: No. some names are very similar and it is hard to tell their difference on the functionality.
HELPFULNESS: Let's think step-by-step. Considering the second rubric concerning the indicative nature of variable names, the review specifically addresses this point by mentioning that some names are very similar, which makes it difficult to discern their functionality. This feedback is constructive as it highlights a problem that can lead to confusion in understanding the code, thus providing the author with a clear aspect to improve upon. Given this context, the HELPFULNESS of the REVIEW, according to the rubric about variable names, can be classified as **"Helpful".**

Figure 2: CoT one-shot example. The designed prompt message and the Chain-of-Thought reasoning in the example are indicated in blue text, whereas the response generated by the model is in yellow.

3.3 RaFI: Feature identification and reasoning

3.3.1 Overview

Chain-of-Thought reasoning offers significant benefits, particularly in understanding the rationale behind assigning the "helpfulness" tag. This approach also encourages deeper thinking among students when providing the evaluation.
However, existing research on peer-assessment helpfulness suggests that determining helpfulness involves multiple factors. Naïvely integrating these factors into a narrative prompt does not guarantee comprehensive coverage of all possible scenarios. To address this issue, we propose the **R**easoning **a**fter **F**eature **I**dentification (**RaFI**) approach.
This method aims to optimize the reasoning step by simulating the process that a human evaluator applies when evaluating helpfulness. This involves initially identifying key

features/ characteristics related to helpfulness within the comments, subsequently determining which of these characteristics contribute to the review's helpfulness, and finally assigning the helpfulness label.

3.3.2 Feature identification

In the field of educational peer assessment, extensive research has been conducted to explore the correlation between various feedback features (characteristics) and helpfulness as perceived by the student, or likelihood of implementing the comments [22, 25]. These studies have identified several key features that contribute to the helpfulness of review comments, including praise [5], problem detection [25], suggestions [37], localization [38], and specificity [4]. In the RaFI framework, we first start by revisiting those features and integrating those definitions into the prompt, then guide the language model to identify those features from the provided comments. A notable gap in previous research is the lack of consideration of the rubric (the rubric items on which the student is invited to comment). This is crucial, as the helpfulness of a comment may depend on the rubric criterion it is responding to. An example of this has been given in Section 3.1.1 (the two comments mentioning the DRY principle). To address this, we introduce two additional criteria for assessing helpfulness in the context of rubric-guided evaluations: "comprehensiveness" and "non-repetitiveness".

We categorize all the features used in the prompt for evaluating the helpfulness into two categories, "essential features" and "constructive features". The detailed explanations of each feature are shown below. The consequential prompt template is shown in Figure 4 (2nd step):

(1) **Essential features**: features that ensure review comments are appropriately guided by the rubric.

- **Comprehensiveness**: This criterion demands that a review should thoroughly address each sub-question outlined in the rubric. A review that violates comprehensiveness is exemplified below, where the response fails to address both parts of the question.
Rubric: Does the design include the test plan? Are the tests shallow or they will really test the functionalities?
Review: Yes, the test plan is included in the project document.

- **Non-repetitiveness**: This ensures that the review offers original responses with valuable information, and avoids merely repeating the language of the rubric. The example below demonstrates a violation of this non-repetitiveness principle by either directly repeating or using synonymous words for the terms outlined in the rubric:
Rubric: Check the commits. Was new code committed during the 2nd round?
Review: No, there were no new commits made during the 2nd round.

(2) **Constructive features**: the review comments may include features that contribute to the helpfulness and potentially assist for improvement.

- **Praise**: Praise is described as a review comment that highlights a positive aspect of a work [25]. Including praise in feedback is commonly recommended for its helpfulness. However, it is only considered certainly helpful when accompanied by detailed explanations and additional valuable feedback components [1]. Here is an example that mentions praise with valuable explanations in the review comments.
Rubric: Have the authors adequately explained the changes to be made to the system?
Review: Yes, they have very clearly mentioned the problems in the project doc that are currently being faced and how they are planning to resolve the issues.

- **Problem statement**: The problem statement is defined as a review comment that identifies an issue of the work that requires attention [15]. The following example illustrates how problem statements can be effectively conveyed with insightful explanations in review comments.
Rubric: If any attributes are missing for Item, please mention which one(s).
Review: The "Teacher" feature is missing when I try to create a course. Seems like the author set it to be invisible.

- **Suggestions**: Suggestion refers to the review comments offering solution in review comments, aimed at fixing a problem or enhancing the quality of the work. Here is an example that illustrates such a suggestion:
Rubric: Do the class diagram and/ or other figures clearly describe the changes to be made to the system??
Review: Figures are concise but good, and provide a good representation of the work to be implemented. However, adding a screenshot of the designed user interface would make it more understandable.

- **Localization**: Localization refers to the review comments pinpointing where the problem occurs [38]. Here is an example:
Rubric: If any attributes are missing for students, mention which one(s).?
Review: The credit card info is missing, which is supposed to be presented on the 2nd row of the student table.

- **Providing Examples**: Providing examples in the review comments refers to the practice where feedback neither explicitly praises nor criticizes but instead offers specific instances or illustrations aligned with the rubric's criteria [4]. Here is an example:
Rubric: Are the commit messages indicative of what changes were made in that commit? Mention examples of where this is not the case.
Review: Commit:246bc8d92c558413d7e4be8f;
Commit:f891c7f5cadf90e3db98fda80e1b9205be2dc6aa.

3.3.3 Reasoning

In the second phase of the RaFI framework, we analyze the helpfulness of reviews by integrating identified features into a logical reasoning process. This approach is illustrated in Figure 3. Our logic indicates that any absence of *essential* features results in an unhelpful review, regardless of the presence of *constructive* features. Informed by prior

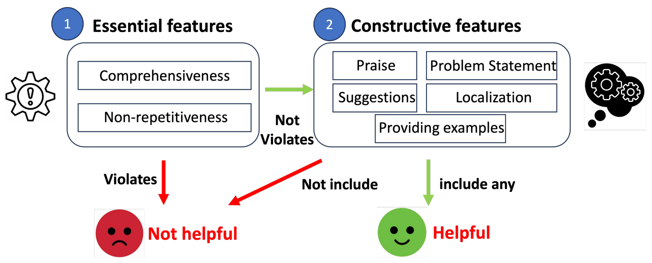
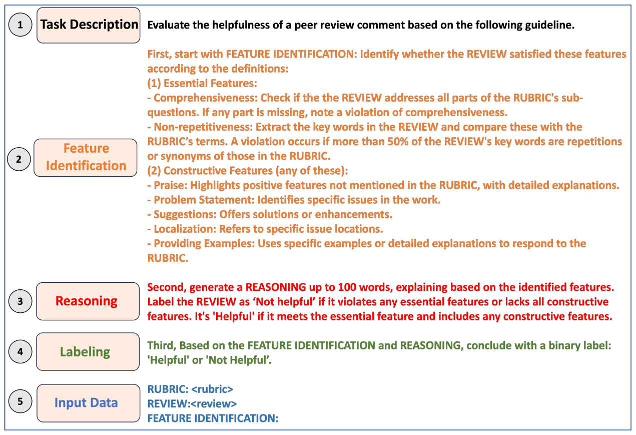


Figure 3: RaFI reasoning process.



1 Task Description
Evaluate the helpfulness of a peer review comment based on the following guideline.

2 Feature Identification
First, start with FEATURE IDENTIFICATION: Identify whether the REVIEW satisfied these features according to the definitions:
(1) Essential Features:
- Comprehensiveness: Check if the the REVIEW addresses all parts of the RUBRIC's sub-questions. If any part is missing, note a violation of comprehensiveness.
- Non-repetitiveness: Extract the key words in the REVIEW and compare these with the RUBRIC's terms. A violation occurs if more than 50% of the REVIEW's key words are repetitions or synonyms of those in the RUBRIC.
(2) Constructive Features (any of these):
- Praise: Highlights positive features not mentioned in the RUBRIC, with detailed explanations.
- Problem Statement: Identifies specific issues in the work.
- Suggestions: Offers solutions or enhancements.
- Localization: Refers to specific issue locations.
- Providing Examples: Uses specific examples or detailed explanations to respond to the RUBRIC.

3 Reasoning
Second, generate a REASONING up to 100 words, explaining based on the identified features. Label the REVIEW as 'Not helpful' if it violates any essential features or lacks all constructive features. It's 'Helpful' if it meets the essential feature and includes any constructive features.

4 Labeling
Third, Based on the FEATURE IDENTIFICATION and REASONING, conclude with a binary label: 'Helpful' or 'Not Helpful'.

5 Input Data
RUBRIC: <rubric>
REVIEW:<review>
FEATURE IDENTIFICATION:

Figure 4: RaFI overall prompt template

research, inclusion of any of them can contribute to a review comment being considered helpful. Hence, a review is classified as helpful if it presents any *constructive* features without lacking both *essential* features. Conversely, it is labeled unhelpful if it omits the *essential* features or lacks all *constructive* features.

4. EXPERIMENTS AND RESULTS

4.1 Dataset

In this paper, we utilize a dataset collected from the Expertiza [9], a peer-assessment tool employed in a master's level computer science course. The system requires students to provide peer assessments of other's work, including both numerical ratings and textual feedback. In particular, all the textual review comments correspond to the pre-defined rubric items specified by the instructor. Our dataset is composed of these textual comments and their associated rubric items, which are essential for conducting our experiments. For evaluation purposes, we collect "helpfulness" tags assigned by authors who receive the review comments, serving as ground-truth labels for assessing our experimental results.

To ensure the quality and reliability of our dataset for both training and evaluation, we also implement quality-control measures. These measures are based on Inter-Rater Reliability (IRR) [16] to filter out data with significant labeling disagreement among team members while assessing the helpfulness. After processing, our final dataset comprises 3,494 labeled entries. Out of these, we extract **500** as a testing dataset for evaluation. The remaining **2,994** entries are utilized for supervised model fine-tuning and served as examples in few-shot prompting scenarios.

4.2 Experiment setting

Large Language Models. This study aims to compare the effectiveness of generative AI models, which require no training data, against traditional supervised fine-tuned models that depend on such training data. We have chosen BERT (340M parameters) [6], DistilBERT (66M parameters) [29], and RoBERTa (355M parameters) [17] as the baseline models. These models are widely recognized for their proficiency in language tasks, particularly in the realm of text classification, which is the fundamental task in peer assessment helpfulness evaluation. For generative AI, we utilize OpenAI's GPT-3.5 Turbo model (20B parameters) [2] for conducting experiments with various prompt templates.

Prompt Template. Considering the inherent complexity of evaluating helpfulness, due to its subjective and ambiguous nature, we initiate our analysis by contrasting the vanilla prompt template (directly asking the model to label the helpfulness) with the Chain-of-Thought (CoT) prompting. This comparison aims to assess the effectiveness of incorporating a reasoning process into the prompt to help with our task. Subsequently, we also evaluate our proposed RaFI framework against traditional CoT prompting. This step is intended to demonstrate the advantages of RaFI's multi-step reasoning, which simplifies complex tasks into more digestible segments for a logical problem-solving process.

Example-Based Prompting. Additionally, we aim to measure the performance improvement achieved by incorporating examples into prompts. Given the complexity of the guidelines, providing a few concrete examples could significantly aid the generative AI in producing the expected responses. Our experiments cover zero-shot prompting (no examples provided), one-shot prompting (a single example provided), and few-shot prompting (four examples provided). We limit the number of examples to avoid exceeding the token limit of the GPT-3.5 model, particularly with lengthy and complex task descriptions.

4.3 Results and discussion

Experimental results are presented in Figure 1. We monitored accuracy, precision, recall, and $f1$ score as the performance metrics. A more detailed analysis of the results will be conducted by addressing the following research questions. To simplify the discussion, comparisons between various models and settings will primarily focus on accuracy and $f1$ score for an informative representation of the overall performance.

RQ1: How do the performance of supervised methods compare with generative AI with prompts, particularly considering the needs for training data?

The result shows that fine-tuning three major supervised language models on our task, with 2,994 labeled training data, yielded modest results. The best performance was achieved by RoBERTa, with an accuracy of 65.23% and an $f1$ score of 56.80%. Although this completely outperforms vanilla prompting in all the scenarios, in one-shot and few-shot prompting, the CoT approach performs similarly to supervised methods. Notably, our RaFI prompting method outperforms the supervised approaches in all scenarios, showing approximately 14% improvement in accuracy and 23% in $f1$ score. This suggests that the generative AI model's advanced language understanding capabilities, combined with well-crafted prompts, can eliminate the need for the training data traditionally required for supervised model fine-tuning. This is particularly advantageous for our task, considering the scarcity of annotated review comments with helpfulness tags.

RQ2: How does the performance vary with different number of examples provided?
The results indicate significant improvements by adding more examples in both vanilla and CoT prompt templates. For instance, adding four examples led to a 12% increase in accuracy for the vanilla template and an impressive 26% increase for the CoT template. However, the RaFI approach showed less improvement, suggesting that the inclusion of more concrete examples is exceptionally beneficial in scenarios where task logic is complex and ambiguous, or when the task description is overly simplistic and allows more randomness in response generation. In contrast, with RaFI's well-defined task description, the benefits of adding examples are more limited (3.9% increase in accuracy and 5.4% increase in $f1$ with four examples).

RQ3: Does Chain-of-Thought (CoT) prompting yield better results with the reasoning process? And how does it perform across different example-provided settings?
In this study, we investigate whether incorporating a reasoning process benefits performance on our task, as it simulates the human process of solving problems step-by-step, promoting a more logical progression of thought compared to direct solution. Our findings reveal that in both one-shot and few-shot scenarios, CoT prompts significantly outperform the vanilla prompts, but are less effective in the zero-shot scenario. This suggests that for complex and domain-specific tasks, clear and explicit guidance from providing examples is essential to encourage deeper and more accurate thinking processes in generative AI models. This insight underscores the importance of tailored concrete examples in the reasoning step for addressing our task.

RQ4: How does our proposed RaFI approach perform in different settings? Does this multi-step reasoning process outperform traditional CoT prompting?
The RaFI approach demonstrated significant improvements over CoT prompting in all settings, underscoring its effectiveness by breaking down the tasks into more manageable segments and providing clearer, more logical guidance. Specifically, in the zero-shot scenario, RaFI achieved 38% higher accuracy and 31% higher $f1$ score compared to CoT, highlighting the effectiveness of optimizing the reasoning by crafting the task description when no examples are provided. Although the performance of CoT improves with more examples included in the few-shot scenario, it still lags behind RaFI, with a 15% lower accuracy and a 12% lower $f1$ score. This suggests that while providing examples can help guide the reasoning process, they may not fully capture the nuances of different inputs, whereas RaFI's multi-step prompts enable a better understanding of both task requirements and context.

Table 1: Experimental results of different settings. The state-of-the-art performance achieved by the supervised fine-tuning model is denoted with *. Our proposed approach is indicated by †. Within each setting, the best result is highlighted in bold. Furthermore, the most outstanding result across all settings is distinguished by a green background.

	Training Data	Accuracy	Precision	Recall	F1
		Supervised Methods			
BERT [6]	2994	62.70%	**88.50%**	35.97%	51.15%
DistilBERT [29]	2994	58.59%	86.84%	**47.12%**	55.27%
RoBERTa [17]*	2994	**65.23%**	87.31%	42.09%	**56.80%**
		Zero-shot Prompting			
Vanilla	-	44.92%	49.22%	45.68%	47.39%
CoT [10]	-	37.70%	43.00%	45.32%	44.13%
RaFI (ours)†	-	**75.78%**	**85.00%**	**67.27%**	**75.10%**
		One-shot Prompting			
Vanilla	-	50.78%	54.04%	62.59%	58.00%
CoT [10]	-	61.52%	61.88%	**75.90%**	68.17%
RaFI (ours)†	-	**77.15%**	**82.33%**	73.74%	**77.80%**
		Few-shot Prompting (k=4)			
Vanilla	-	56.45%	56.66%	**84.17%**	67.73%
CoT [10]	-	64.45%	62.97%	83.81%	71.91%
RaFI (ours)†	-	**79.69%**	**84.25%**	76.98%	**80.45%**

5. CONCLUSION

In this paper, we explore the use of generative AI models for evaluating the helpfulness of peer assessment comments, highlighting their ability to adapt to tasks without training data usually required for supervised language model. Our study conducts experiments with prompt engineering techniques, such as Chain-of-Thought and few-shot prompting, to improve the model's output. A key contribution of our work is the development of the **R**easoning **a**fter **F**eature **I**dentification (RaFI) framework, as an optimized reasoning process to craft the prompt and lead to more precise and desired responses. Significantly, RaFI outperforms both vanilla and CoT prompts in many settings and shows superiority over three predominant supervised language models, without needing any training data. This research indicates that generative AI, when combined with crafting prompt design, can be an effective substitute for traditional fine-tuning methods in evaluating review-comment helpfulness, thereby eliminating the reliance on training datasets. Future work will focus on applying RaFI framework to various other domains and tasks to assess its effectiveness.

6. REFERENCES

[1] J. L. Bienstock, N. T. Katz, S. M. Cox, N. Hueppchen, S. Erickson, E. E. Puscheck, et al. To the point: medical education reviews—providing feedback. *American journal of obstetrics and gynecology*, 196(6):508–513, 2007.

[2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages

1877–1901. Curran Associates, Inc., 2020.

[3] Y. Chae and T. Davidson. Large language models for text classification: From zero-shot learning to fine-tuning. *Open Science Foundation*, 2023.

[4] K. Cho. Machine classification of peer comments in physics. In *Educational Data Mining 2008*, 2008.

[5] K. Cho, C. D. Schunn, and D. Charney. Commenting on writing: Typology and perceived helpfulness of comments from novice peer reviewers and subject matter experts. *Written communication*, 23(3):260–294, 2006.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[7] K. S. Double, J. A. McGrane, and T. N. Hopfenbeck. The impact of peer assessment on academic performance: A meta-analysis of control group studies. *Educational Psychology Review*, 32:481–509, 2020.

[8] D. Gamage, T. Staubitz, and M. Whiting. Peer assessment in moocs: Systematic literature review. *Distance Education*, 42:268–289, 2021.

[9] E. Gehringer, L. Ehresman, S. G. GConger, and P. Wagle. Reusable learning objects through peer review: The expertiza approach. volume 3, pages 1–2, 2006.

[10] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa. Large language models are zero-shot reasoners. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc., 2022.

[11] S. Lai, L. Xu, K. Liu, and J. Zhao. Recurrent convolutional neural networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

[12] A. K. Lampinen, I. Dasgupta, S. C. Chan, K. Matthewson, M. H. Tessler, A. Creswell, J. L. McClelland, J. X. Wang, and F. Hill. Can language models learn from explanations in context? *arXiv preprint arXiv:2204.02329*, 2022.

[13] H. Li, Y. Xiong, C. V. Hunter, X. Guo, and R. Tywoniw. Does peer assessment promote student learning? a meta-analysis. *Assessment and Evaluation in Higher Education*, 45:193–211, 6 2020.

[14] X. L. Li and P. Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[15] C. Liu, J. Cui, R. Shang, Y. Xiao, Q. Jia, and E. Gehringer. Improving problem detection in peer assessment through pseudo-labeling using semi-supervised learning. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 391–397, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

[16] C. Liu, D. Doshi, M. Bhargava, R. Shang, J. Cui, D. Xu, and E. Gehringer. Labels are not necessary: Assessing peer-review helpfulness using domain adaptation based on self-training. In E. Kochmar, J. Burstein, A. Horbach, R. Laarmann-Quante, N. Madnani, A. Tack, V. Yaneva, Z. Yuan, and T. Zesch, editors, *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 173–183, Toronto, Canada, July 2023. Association for Computational Linguistics.

[17] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[18] L. Loukas, I. Stogiannidis, P. Malakasiotis, and S. Vassos. Breaking the bank with ChatGPT: Few-shot text classification for finance. In C.-C. Chen, H. Takamura, P. Mathur, R. Sawhney, H.-H. Huang, and H.-H. Chen, editors, *Proceedings of the Fifth Workshop on Financial Technology and Natural Language Processing and the Second Multimodal AI For Financial Forecasting*, pages 74–80, Macao, 20 Aug. 2023. -.

[19] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics.

[20] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao. Deep learning–based text classification: a comprehensive review. *ACM computing surveys (CSUR)*, 54(3):1–40, 2021.

[21] N. V. Nayak, P. Yu, and S. H. Bach. Learning to compose soft prompts for compositional zero-shot learning. *arXiv preprint arXiv:2204.03574*, 2022.

[22] M. M. Nelson and C. D. Schunn. The nature of feedback: How different types of peer feedback affect writing performance. *Instructional Science*, 37:375–401, 7 2009.

[23] M. M. Patchan and C. D. Schunn. Understanding the benefits of providing peer feedback: how students respond to peers' texts of varying quality. *Instructional Science*, 43:591–614, 2015.

[24] M. M. Patchan, C. D. Schunn, and R. J. Clark. Accountability in peer assessment: examining the effects of reviewing grades on peer ratings and peer feedback. *Studies in Higher Education*, 43:2263–2278, 12 2018.

[25] M. M. Patchan, C. D. Schunn, and R. J. Correnti. The nature of feedback: how peer feedback features affect students' implementation rate and quality of revisions. *Journal of Educational Psychology*, 108:1098–1120, 11 2016.

[26] G. Qin and J. Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*, 2021.

[27] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised

multitask learners.

[28] L. Reynolds and K. McDonell. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI EA '21, New York, NY, USA, 2021. Association for Computing Machinery.

[29] V. Sanh, L. Debut, J. Chaumond, and T. Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[30] V. Sanh, A. Webson, C. Raffel, S. H. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, T. L. Scao, A. Raja, et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*, 2021.

[31] X. Sun, X. Li, J. Li, F. Wu, S. Guo, T. Zhang, and G. Wang. Text classification via large language models. pages 8990–9005. Association for Computational Linguistics, 12 2023.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[33] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[34] J. Wei, X. Wang, D. Schuurmans, M. Bosma, b. ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022.

[35] K. Weiss, T. M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

[36] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt. A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382*, 2023.

[37] Y. Xiao, T. Wang, X. Sun, Y. Li, Y. Song, J. Cui, Q. Jia, C. Liu, and E. F. Gehringer. Modeling review helpfulness with augmented transformer neural networks. pages 83–90, 2022.

[38] W. Xiong and D. Litman. Identifying problem localization in peer-review feedback. In *Intelligent Tutoring Systems: 10th International Conference, ITS 2010, Pittsburgh, PA, USA, June 14-18, 2010, Proceedings, Part II 10*, pages 429–431. Springer, 2010.

[39] G. Zingle, B. Radhakrishnan, Y. Xiao, E. Gehringer, Z. Xiao, F. Pramudianto, G. Khurana, and A. Arnav. Detecting suggestions in peer assessments. *International Educational Data Mining Society*, 2019.

[40] Y. D. Çevik, T. Haşlaman, and S. Çelik. The effect of peer assessment on problem solving skills of prospective teachers supported by online learning activities. *Studies in Educational Evaluation*, 44:23–35, 2015.