Debiasing In-Context Learning by Instructing LLMs How to Follow Demonstrations

Lvxue Li[1,3], **Jiaqi Chen**[1,3], **Xinyu Lu**[1,3], **Yaojie Lu**[1*], **Hongyu Lin**[1],
Shuheng Zhou[5], **Huijia Zhu**[5], **Weiqiang Wang**[5],
Zhongyi Liu[5], **Xianpei Han**[1,2,4], **Le Sun**[1,2,4]
[1]Chinese Information Processing Laboratory, [2]State Key Laboratory of Computer Science
Institute of Software, Chinese Academy of Sciences, Beijing, China
[3]University of Chinese Academy of Sciences, Beijing, China
[4]Key Laboratory of System Software, Chinese Academy of Sciences
[5]Ant Group
{lilvxue2021,luyaojie,hongyu,xianpei,sunle}@iscas.ac.cn
{shuheng.zsh,weiqiang.wwq,zhongyi.lzy}@antgroup.com
huijia.zhj@antfin.com

Abstract

In-context learning(ICL) has gained considerable attention due to its data efficiency and task adaptability. Unfortunately, ICL suffers from the demonstration bias, i.e., its performance and robustness are severely affected by the selection and ordering of demonstrations. In this paper, we identify that such demonstration bias may primarily stem from the semantic ambiguity induced by demonstrations, i.e., a demonstration may indicate multiple input-to-label mappings and its mapping can be interpreted differently in different contexts by LLMs. Such semantic ambiguity disrupts task comprehension during ICL and results in performance fluctuations. To resolve the semantic ambiguity problem, this paper further proposes two de-biasing strategies to mitigate demonstration bias in in-context learning. Experiments on six datasets show that our methods can effectively alleviate demonstration bias and significantly improve task performance.

1 Introduction

In-context learning(ICL) has gained considerable attention in recent years, wherein a LLM can perform an unseen task by only conditioning on several in-context input-output demonstrations (Brown et al., 2020). Due to its minimal data requirements and zero parameter updates, ICL enables developers to efficiently and flexibly apply LLMs in different domains (King and Flanigan, 2023; Gero et al., 2023; Huang et al., 2023; Winata et al., 2023).

Despite its effectiveness and popularity, recent studies have highlighted that ICL is highly sensitive to the selection and order of demonstrations (Zhao et al., 2021; Lu et al., 2022) – which this paper calls *demonstration bias*. Such a demonstration bias often results in significant performance fluctuations and severely undermines the robustness of LLMs. Although many works have been proposed to search and generate optimal demonstrations for real-world tasks, it is still unclear what the underlying reasons of the demonstration bias in ICL are (Liu et al., 2022; Sorensen et al., 2022; Gonen et al., 2022; Li and Qiu, 2023a; Wang et al., 2023).

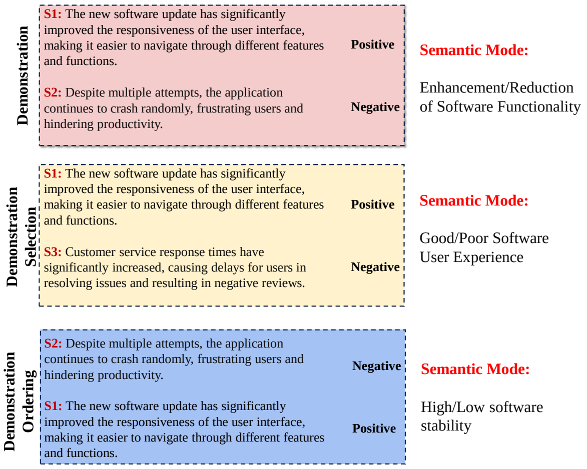

Demonstration
S1: The new software update has significantly improved the responsiveness of the user interface, making it easier to navigate through different features and functions.
Positive
S2: Despite multiple attempts, the application continues to crash randomly, frustrating users and hindering productivity.
Negative
Semantic Mode:
Enhancement/Reduction of Software Functionality
Demonstration Selection
S1: The new software update has significantly improved the responsiveness of the user interface, making it easier to navigate through different features and functions.
Positive
S3: Customer service response times have significantly increased, causing delays for users in resolving issues and resulting in negative reviews.
Negative
Semantic Mode:
Good/Poor Software User Experience
Demonstration Ordering
S2: Despite multiple attempts, the application continues to crash randomly, frustrating users and hindering productivity.
Negative
S1: The new software update has significantly improved the responsiveness of the user interface, making it easier to navigate through different features and functions.
Positive
Semantic Mode:
High/Low software stability

Figure 1: Illustrative examples of demonstration bias in In-Context Learning. Various demonstration organizations (left) can significantly influence the semantic modes chosen by LLMs (right).

In this paper, we identify that such a demonstration bias may primarily stem from the semantic ambiguity induced by demonstrations, i.e., a demonstration may indicate multiple input-to-label mappings(this paper refers to the possible input-to-label mappings of a demonstration as its semantic modes) and its semantic modes can be interpreted differently in different contexts by LLMs. For in-

[*] Corresponding authors.

Findings of the Association for Computational Linguistics: ACL 2024, pages 7203–7215
August 11-16, 2024 ©2024 Association for Computational Linguistics

stance, as shown in Figure 1, the demonstration S1 has three different semantic modes: (1) Enhancement/Reduction of Software Functionality; (2) Good/Poor Software User Experience; and (3) High/Low software stability. LLMs may interpret it differently when accompanying different other demonstrations. It is obvious that when demonstrations show high semantic ambiguity, an LLM will have difficulty selecting the appropriate semantic mode of a task, causing sensitivity and bias in the demonstration organization in ICL. For example, given the new input "Despite the implementation of new security measures, users report frequent authentication errors and difficulties accessing their accounts, leading to usability issues," an LLM might classify it as Positive under the mode "Enhancement/Reduction of Software Functionality," but as Negative under the mode "Good/Poor Software User Experience."

To investigate the impact of demonstration ambiguity, we conduct comprehensive experiments on varying degrees of semantic ambiguity, across different models and datasets. Specifically, we first design a *semantic ambiguity score* which can evaluate the divergence of a demonstration's semantic mode across various contexts. In this way, a low ambiguity score indicates a demonstration will have a stable semantic mode and therefore is less likely to be interpreted differently in differing contexts by LLMs. Based on the above measure, our findings revealed a strong correlation between the semantic ambiguity of demonstration and the performance fluctuation of ICL. That is, given a demonstration, as its semantic ambiguity increases, it is more difficult for LLMs to select the correct semantic modes in in-context learning, which in turn leads to greater instability in performance.

Based on the above findings, we further propose two de-biasing strategies for in-context learning, named Instance-Free Demonstration Reordering and Self-Explanatory In-Context Learning, which can effectively help LLMs accurately select semantic modes and thus significantly reduce the demonstration bias. First, we propose a *Instance-Free Demonstration Reordering* method, which progressively selects demonstrations by maximizing the semantic ambiguity reduction of in-context demonstrations. Second, we present the *Self-Explanatory In-Context Learning framework*, which generates explicit explanatory guidelines for each instance and then instructs LLMs to select appropriate semantic modes by following these guidelines. This self-explanatory mechanism enables LLMs to reflect their internal thinking, reasoning, and decision guidelines, and these explanatory guidelines can additionally instruct LLMs the semantic mode demonstrations want to convey. By instructing LLMs with better demonstration order and self-explanatory guidelines, LLMs can effectively address the semantic ambiguity problem and significantly reduce the demonstration bias of ICL. We conducted experiments on six classification datasets, and the results verified the effectiveness of our methods.

In summary, our contributions are as follows:

1. We identify that the demonstration bias may primarily stem from the semantic ambiguity induced by demonstrations and reveal a strong correlation between the semantic ambiguity of demonstration and the performance fluctuation of ICL.

2. We propose two de-biasing strategies for in-context learning, named Instance-Free Demonstration Reordering and Self-Explanatory In-Context Learning, which can effectively help LLMs to accurately select semantic modes and thus significantly reduce the demonstration bias.

3. Experimental results demonstrate the effectiveness of our methods in significantly reducing the demonstration bias and enhancing performance of ICL.

2 Semantic Ambiguity in In-Context Learning

In this section, we explore the impact of semantic ambiguity within demonstrations on in-context learning. In §2.1, we introduce the notion of a semantic ambiguity score to quantify the level of comprehension exhibited by LLMs for a given demonstration. Then we conduct experiments to further validate our hypothesis in §2.2 and §2.3.

2.1 Semantic Ambiguity Score of Demonstrations

To validate our hypothesis as outlines in §1, we propose the concept of a *semantic ambiguity score* to assess the consistency of LLM's comprehension of a given demonstration d_i across different contexts:

$$A_i = \sum_{k=1}^{N} |P_k(d_i d_u) - P_k(d_u d_i)| \tag{1}$$

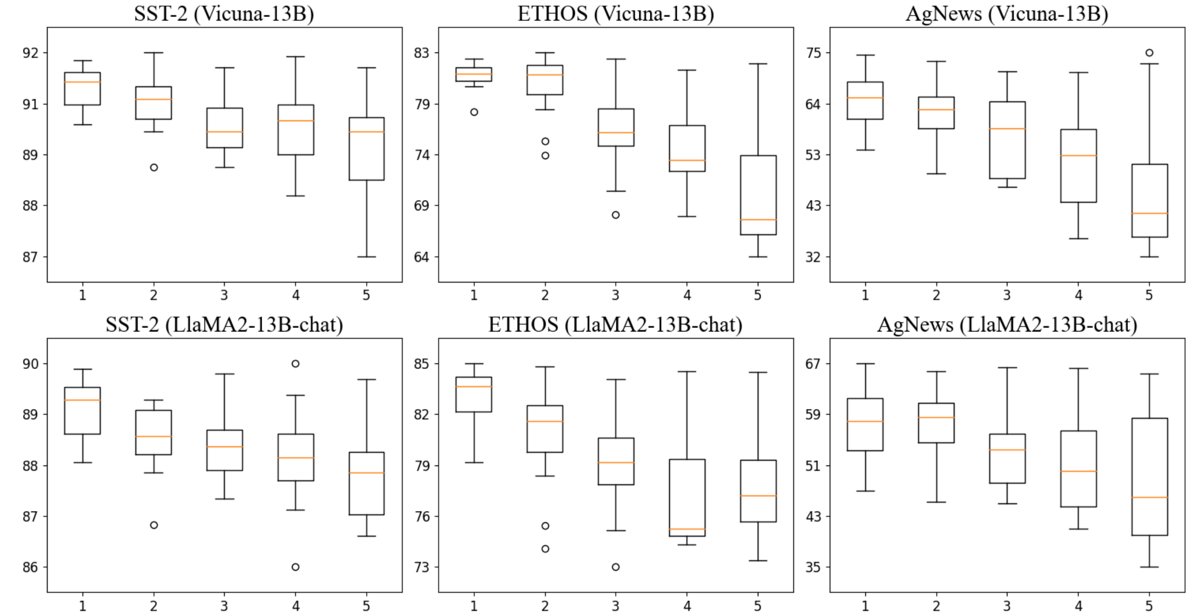


Figure 2: Semantic Ambiguity Results on Vicuna-13B and LlaMA2-13B-chat. The horizontal axis represents the demonstration grouping, where Group 1 corresponds to the lowest ambiguity, and Group 5 corresponds to the highest. The vertical axis represents accuracy. Each box-plot reports the aggregated results of 20 randomly shuffled trials.

We initially introduce an uninformative demonstration d_u = ("None", "None") to prevent the introduction of task-irrelevant semantic modes. To systematically assess the model's comprehension of d_i across various contexts, we interchange d_i and d_u, creating two contexts formally distinct but conveying identical information: $C_1 = d_i d_u$ and $C_2 = d_u d_i$. Furthermore, we introduce the concept of **Input Label Probability** P_k to aid in encapsulating the LLM's understanding of a given context C. This involves introducing an additional demonstration, denoted as $d_k = (x_k, y_k)$, and leveraging the LLM's label probability to gauge its extracted semantic mode from the given context C:

$$P_k(C) = P_{LLM}(y_k \mid C, x_k) \tag{2}$$

In our approach, we tokenize y_k and extract the probability assigned to the first token. For example, in SST-2 dataset with candidate labels such as *Positive* or *Negative*, we derive the probabilities P("_Pos") or P("_Neg") as the input label probability. To analyze the impact of d_i, we systematically traverse through all d_k in our demonstration pool $D = \{d_i\}^N$, ensuring that $y_i = y_k$.

Based on the above definition, we hypothesize that demonstrations characterized by lower semantic ambiguity scores yield more consistent interpretations by LLMs, thereby reducing their susceptibility to variations across diverse contexts.

2.2 Experiments

We use **SST-2** (Socher et al., 2013) for sentiment analysis, **ETHOS** (Mollas et al., 2020) for hate speech detection, and **AgNews** (Zhang et al., 2015) for topic classification[1]. We choose Vicuna-13B (Zheng et al., 2023) and LlaMA2-13B-chat (Touvron et al., 2023) as our primary model.

We begin by randomly selecting $N \times M$ demonstrations from the training split for each dataset. Here, $N = 25$ represents the number of demonstrations in our pool, while M indicates the number of candidate labels in each dataset. For clarity, the SST2 dataset presents $M = 2$ labels: "Positive" and "Negative," whereas the AgNews dataset encompasses $M = 4$ distinct labels.

Following this selection, we leverage Eq. 1 to compute the semantic ambiguity scores for demonstrations. For each of the M candidate labels, we arrange the demonstrations associated with that label in ascending order according to their scores and evenly distribute them into five groups. Consequently, each of the five groups comprises $5M$ demonstrations, maintaining a balanced distribution of candidate labels. This strategy guarantees a

[1] We use the version on HuggingFace (Lhoest et al., 2021) of all datasets in this paper.

fair and thorough assessment.

For evaluation, we randomly selected 300 instances from the test split, employing the previously mentioned groups as demonstrations. Subsequently, the results are averaged across 20 random shuffling within each group.

2.3 Results and Analysis

Figure 2 demonstrates the results of our semantic ambiguity experiments on Vicuna-13B and LlaMA2-13B-chat across three different tasks. We can see that:

1) **Semantic ambiguity presents challenges for LLMs in selecting correct semantic modes, resulting in fluctuations in ICL performance.** As illustrated in Figure 2, the performance of in-context learning varies across all demonstration groups, irrespective of low or high ambiguity score levels. This supports our hypothesis that semantic ambiguity is pervasive in demonstrations, thereby influencing LLMs in selecting the appropriate semantic modes. Furthermore, we observe a direct proportional relationship observed between the ambiguity score of demonstrations and performance variance. For instance, on Vicuna-13B, the standard deviation for the first set in the ETHOS dataset is only 1.07. In contrast, for the fifth set, characterized by a heightened ambiguity score, the deviation significantly increases to 5.12. This indicates that demonstrations with higher ambiguity scores exert a greater influence on LLM's comprehension of demonstrations. LLMs struggle to extract correct semantic modes relevant to the task, resulting in increased variances in ICL performance.

2) **The performance of in-context learning depends on semantic modes in demonstrations.** We have uncovered a fascinating phenomenon: as the ambiguity score of demonstrations gradually increases, the best performance among various shufflings remains competitive with the optimal group. For instance, on LlaMA2-13B-chat, the performance of the fifth group on AgNews fluctuates by approximately 30 percentage points, while the best ordering achieves an accuracy of 65.6, which competes closely with the first group's 67.2 accuracy. We propose that the observed phenomenon may stem from the relationship between conveyed information and the level of ambiguity in demonstrations. Demonstrations with higher ambiguity can prompt LLMs to access a wider range of potential semantic modes. As long as a correct mode exists within the demonstration, there is a chance that the LLM can extract it and achieve an optimal result, albeit with greater difficulty.

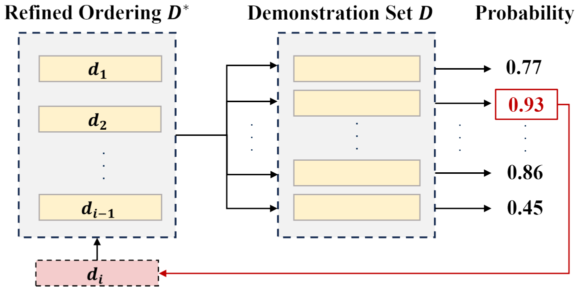


Figure 3: An overview of our demonstration reordering method. Given We use probability (as shown above) or entropy as objectives to search for the next demonstration d_i.

3 Instructing Large Language Models to Follow In-Context Demonstrations

In §2.3, we demonstrate that semantic ambiguity within demonstrations poses challenges for large language models to extract appropriate semantic modes, affecting their capacity to fully comprehend internal tasks and resulting in fluctuations in in-context learning performance. Expanding on these insights, this section elucidates our proposed methods designed to help LLMs improve capabilities to select correct semantic modes and thus mitigate demonstration bias. First, we propose *Instance-Free Demonstration Reordering* method in §3.1, then we present the *Self-Explanatory In-Context Learning* framework in §3.2.

3.1 Instance-Free Demonstration Reordering

Drawing inspiration from Section 2.3, we propose that tailoring demonstrations to minimize semantic ambiguity can significantly reduce their impact on LLMs, thus enhancing the likelihood of LLMs selecting the correct semantic modes. Based on this premise, we introduce a demonstration reordering method, outlined in Fig 3. Given a demonstration set D, we meticulously traverse it step by step to maximize the reduction of semantic ambiguity. At step i, based on the previously searched ordering $D^* = (d_1, d_2, \cdots, d_{i-1})$, we select the next demonstration d_i from D according to a predefined metric and update it into D^*.

Given the high time complexity involved in calculating the ambiguity score as detailed in Equation 1, efficiency bottlenecks can arise in practical applications. To mitigate this issue, we propose using

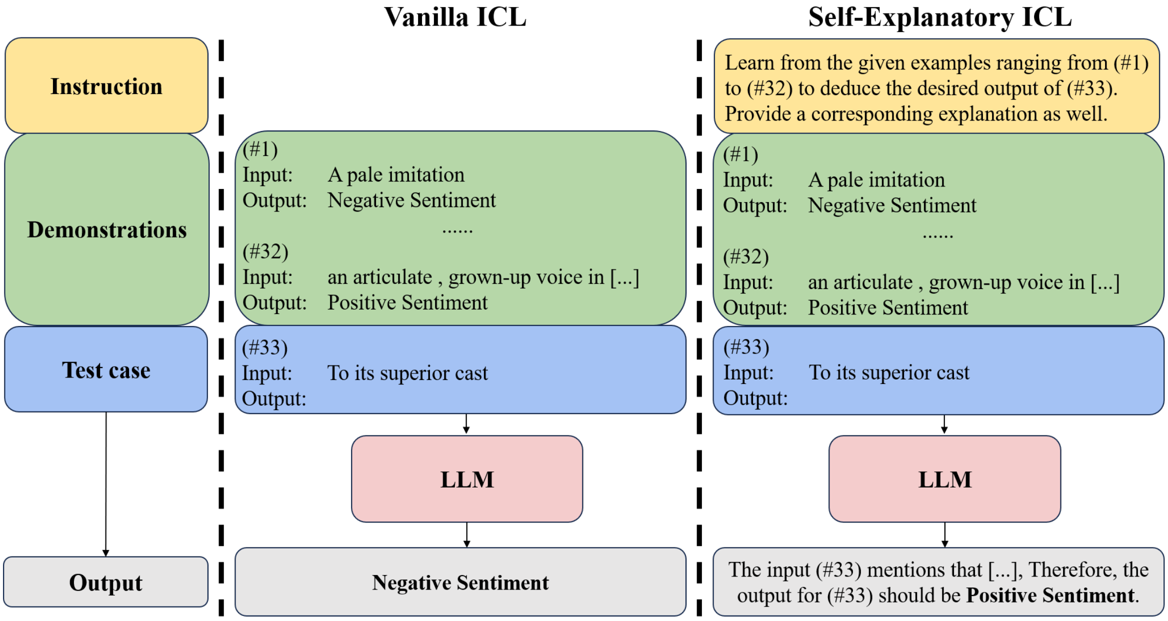

Vanilla ICL
Self-Explanatory ICL
Instruction
Demonstrations
Test case
Output
Learn from the given examples ranging from (#1) to (#32) to deduce the desired output of (#33). Provide a corresponding explanation as well.
(#1)
Input: A pale imitation
Output: Negative Sentiment
......
(#32)
Input: an articulate , grown-up voice in [...]
Output: Positive Sentiment
(#33)
Input: To its superior cast
Output:
LLM
Negative Sentiment
The input (#33) mentions that [...], Therefore, the output for (#33) should be Positive Sentiment.

Figure 4: An overview of our self-explanatory ICL framework. Compared with vanilla ICL, self-explanatory ICL adds a self-explanatory instruction into the model inputs, stimulating the ability of the model to generate explanatory guidelines and the task result at the same time.

the probability or entropy of subsequent demonstrations as our metric. This alternative allows us to assess the LLM's understanding of the task and its confidence in extracting semantic modes. As outlined below, we have developed three distinct metrics:

1. Probability-Candidate: We aggregate the probability of the label within the model across all candidate labels (e.g. $P(\text{"Positive"}) + P(\text{"Negative"})$ for SST-2) to derive the *Candidate Label Probability*. By maximizing this probability of the next demonstration, we can select the demonstration that optimizes task clarity.

$$d_i \sim \arg\max_{d \in D} P_c(d \mid d_1, d_2, \ldots, d_{i-1}) \quad (3)$$

2. Probability-Gold: Following the setting outlined in Section 2, we leverage the probability of the gold label within the upcoming demonstration, defining it as the *Gold Label Probability*. By optimizing this metric, we aim to choose the demonstration that maximizes the likelihood of the LLM identifying the correct semantic interpretation.

$$d_i \sim \arg\max_{d \in D} P_g(d \mid d_1, d_2, \ldots, d_{i-1}) \quad (4)$$

3. Entropy: We leverage label probability entropy to establish the *Label Entropy*. By minimizing this metric, we aim to identify the demonstration that optimizes the model's confidence in selecting semantic modes.

$$d_i \sim \arg\min_{d \in D} E(d \mid d_1, d_2, \ldots, d_{i-1}) \quad (5)$$

This approach navigates through candidate demonstrations, sequentially seeking the optimal ones. After undergoing $n = |D|$ iterations, we arrive at a novel arrangement of D.

To mitigate the risk of converging toward local optima, we employ a strategy akin to beam search, preserving the top-5 candidate demonstration sets at every step of the search process. As a result, we retain the top-1 sequence and utilize it as the final demonstration.

3.2 Self-Explanatory In-Context Learning

As delineated in 2.3, whenever an appropriate semantic mode is present within a given demonstration, LLMs stand a chance of extracting it, albeit with increased difficulty. We propose that providing guidelines for LLMs can aid in enhancing the understanding of demonstrations and consequently improve the ability to extract accurate semantic modes.

In this study, we introduce a self-explanatory in-context learning framework, illustrated in Figure 4.

Unlike conventional in-context learning described in Equation 6, our innovative framework integrates a tailored input representation denoted as $X = [I, D, x_{test}]$. Here, I acts as a self-explanatory instruction explicitly guiding the model's attention towards the information conveyed in demonstrations. This strategy aids LLMs in generating instance-level explanations E_{test} that reflect its internal thinking, reasoning, and decision-making processes. Consequently, it guides LLMs in capturing the semantic mode expressed by demonstrations and generating the final answer.

$$y_{test} \sim P_{LLM}(y \mid D, x_{test}) \tag{6}$$

$$E_{test}, y_{test} \sim P_{LLM}(y \mid I, D, x_{test}) \tag{7}$$

In our framework, demonstrations in D lack explanations, which contradicts our goal of incorporating explanatory elements in the model's output. This structural mismatch presents a notable challenge in crafting self-explanatory instructions. To overcome this obstacle and gain better control over the model's generated results, we utilize the Optimization by PROmpting (OPRO, (Yang et al., 2023)) method to refine our instructions, leveraging its effectiveness and versatility in practical scenarios. Details are outlined in Appendix A.

4 Experiments

4.1 Dataset and Evaluation

Following previous work (Wei et al., 2023b; Sun et al., 2023; Gu et al., 2023), we conduct experiments on six classification datasets: We use **SST-2** (Socher et al., 2013), **FP** (Malo et al., 2013), **IMDB** (Maas et al., 2011) and **CR** (Ding et al., 2008) for sentiment analysis, **ETHOS** (Mollas et al., 2020) for hate speech detection, and **AgNews** (Zhang et al., 2015) for topic classification.

To conduct a comprehensive evaluation, we use 20 different shufflings for each set of randomly sampled demonstrations and use 10 different sets for each experiment, giving a total of 200 trials. Results are presented in terms of mean accuracy score and standard deviation drawing from 500 instances from the test split of each dataset. We also ensure an equal distribution of candidate labels in demonstrations.

4.2 Implementations

We assess the effectiveness of our methodologies across both closed-source and open-source large language models. For closed-source model, specifically ChatGPT (gpt-3.5-turbo-0613), we employ a sampling decoding strategy, configuring the temperature to 0.75 and the top_p value to 0.9. For open-source models, our evaluation encompasses Vicuna-13B (Zheng et al., 2023), Vicuna-7B, and LlaMA2-13B-chat (Touvron et al., 2023), for which we utilize a greedy decoding strategy. This configuration enables us to ascertain the adaptability of our methodologies to compact-sized LLMs.

For both the vanilla ICL baseline and our proposed demonstration reordering method, we established a maximum output length of 128 tokens. For our self-explanatory ICL variant, we extended the maximum output length to 896 tokens, ensuring complete generation of explanations. Generally, the output of out self-explanatory ICL method adheres to the format of "[Explanation] [Label]" owing to our utilization of instruction-tuned models. Therefore, we employ regular expression matching to identify the candidate labels within the generated output, scanning from right to left.

4.3 Overall Results

We conduct experiments on all six datasets using Vicuna-13B to ensure the effectiveness of our methods on different tasks. As shown in Table 1, we can see that:

1) **Demonstration bias commonly exists in in-context learning.** Table 1 unveils considerable variability in performance of vanilla in-context learning. Particularly across the IMDB dataset, the standard deviation stands at 12.95, with accuracy spanning from a minimum of 47.6 to a maximum of 92.6. When coupled with our observations illustrated in Figure 2, we propose that demonstration bias significantly influences the capability of LLMs to accurately select semantic modes, thereby causing fluctuations in ICL performance.

2) **Our methods excel in mitigating demonstration bias across six datasets.** As illustrated in Table 1, our proposed methods consistently achieve superior performance and heightened robustness compared to vanilla ICL across all datasets. Specifically, our demonstration reordering method consistently identifies orderings that achieve optimal performance across different sampled demonstrations. On the IMDB dataset, all three metrics we employed for searching surpass the baseline by an average of 4-5 percentage points across 10 different selections. Additionally, we observed that our self-explanatory in-context learning framework sig-

Datasets	**SST-2**	**ETHOS**	**FP**	**IMDB**	**AgNews**	**CR**
Vanilla ICL	87.43 / 3.26	82.03 / 3.14	88.47 / 4.29	74.09 / 12.95	76.22 / 5.50	74.56 / 11.92
Reordering with Probability-Candidate	88.98 / **1.32**	82.98 / 1.95	89.70 / 2.04	76.74 / 10.78	77.96 / 1.77	**79.36** / 7.50
Reordering with Probability-Gold	89.36 / 2.10	83.52 / 2.28	**89.88 / 1.94**	**77.22** / 10.74	**78.84 / 1.20**	78.22 / 7.60
Reordering with Entropy	**89.56** / 2.18	**83.78** / 1.65	89.64 / 2.50	76.76 / **9.79**	77.90 / 1.80	76.24 / **7.27**
Self-Explanatory ICL	89.04 / 1.75	82.80 / **1.40**	89.64 / 2.08	74.22 / 12.28	78.12 / 2.42	73.00 / 8.66

Table 1: Results (mean / std.) of different in-context learning strategies with the Vicuna-13B backbone. To adhere to the maximum input length constraints of the Vicuna-13B model, we utilize a 4-shot approach for IMDB and CR, 8-shot for ETHOS, 16-shot for FP and AgNews, and 32-shot for SST-2. Baseline results are averaged over 10 distinct samplings across 20 random shufflings. Our method's results are averaged over 10 distinct samplings.

Datasets	**SST-2**	**ETHOS**	**AgNews**
Vanilla ICL	65.54 / 10.84	59.06 / 6.25	32.94 / 3.40
Probability-Candidate	74.00 / 6.60	71.04 / 6.10	35.02 / 2.96
Probability-Gold	69.30 / 9.34	67.90 / 5.78	**35.98** / 3.10
Entropy	68.84 / 7.27	67.06 / 4.84	34.90 / **2.16**
Self-Explanatory ICL	**83.40 / 3.86**	**75.02 / 2.88**	34.54 / 2.96

Table 2: Results (mean / std.) of different in-context learning strategies with the Vicuna-13B backbone under semantical unrelated-labels settings.

Models	**Vicuna-7B**	**LlaMA2-13B**	**GPT-3.5-turbo**
Vanilla ICL	57.74 / 14.28	79.92 / 4.14	88.60 / 2.46
Reordering Method			
Probability-Candidate	68.98 / 8.87	84.64 / 1.36	-
Probability-Gold	75.08 / 2.21	84.40 / 2.21	-
Entropy	74.88 / 3.51	84.72 / 2.90	-
Self-Explanatory ICL			
Exps of Vicuna-7B	70.50 / 6.63	73.30 / 5.37	80.20 / 3.65
Exps of LlaMA2-13B	72.10 / 4.85	81.14 / 1.85	84.40 / 2.25
Exps of GPT-3.5-turbo	87.90 / 1.58	86.70 / 3.61	88.70 / 1.90

Table 3: Results (mean / std.) of different in-context learning strategies on AgNews with different LLMs. For self-explanatory ICL, we present performance metrics guided by Exps (explanations) provided by diverse LLMs.

nificantly enhances robustness across all datasets while maintaining competitive or superior performance compared to the baseline. This indicates that by progressively searching for demonstrations that maximize semantic ambiguity reduction or instructing LLMs in generating self-explanatory guidelines, our methods effectively help LLMs extract correct semantic modes from demonstrations.

4.4 Results of Semantic Unrelated-Label Settings

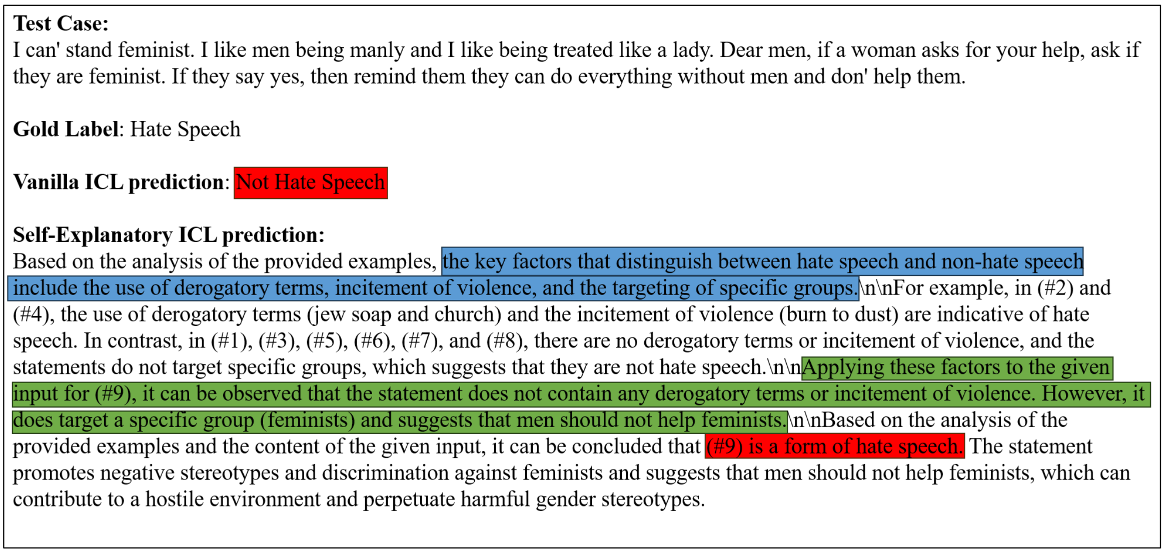
Pan et al. (2023) dissect the in-context learning ability of LLMs into two distinct components: task recognition, which distinguishes tasks from demonstrations and leverages the pre-trained priors of LLMs, and task learning, which learns input-to-label mappings based on demonstrations. We contend that the greater the task learning ability, the more accurately the model captures semantic modes. To support this assertion, we conduct supplementary *Semantic Unrelated-Labels ICL* experiments, providing evidence that our methods enhance LLMs' task learning ability.

In this configuration, labels associated with demonstration instances are intentionally transformed into task-unrelated terms following a predefined mapping, such as substituting "Positive" with "Foo" and associating "Negative" with "Bar." This adjustment aims to eliminate semantic biases from demonstrations, compelling LLMs to rely solely on acquiring input-to-label mappings. Table 2 demonstrates the performance of Vicuna-13B across three distinct tasks. We can see that:

Our method can significantly improve the capability of large language models to accurately capture semantic modes. It is evident that our methodology consistently surpasses the vanilla baseline across varied tasks. Specifically, within the ETHOS dataset, our methods improve upon the baseline by approximately 10 percentage points while notably enhancing robustness. This underscores the effectiveness of our approach in enhancing the model's capability in extracting correct semantic modes within demonstrations, thereby enhancing its in-context learning performance.

4.5 Results of Different Large Language Models

We perform experiments on the AgNews dataset employing Vicuna-7B, LlaMA2-13B-chat, and ChatGPT to substantiate the transferability of our methodologies. Furthermore, to assess the quality of explanations produced by various models, we leverage explanations from one model as a guiding reference for others. As depicted in Table 3, our findings illustrate that:



Test Case:
I can' stand feminist. I like men being manly and I like being treated like a lady. Dear men, if a woman asks for your help, ask if they are feminist. If they say yes, then remind them they can do everything without men and don' help them.

Gold Label: Hate Speech

Vanilla ICL prediction: Not Hate Speech

Self-Explanatory ICL prediction:
Based on the analysis of the provided examples, the key factors that distinguish between hate speech and non-hate speech include the use of derogatory terms, incitement of violence, and the targeting of specific groups.\n\nFor example, in (#2) and (#4), the use of derogatory terms (jew soap and church) and the incitement of violence (burn to dust) are indicative of hate speech. In contrast, in (#1), (#3), (#5), (#6), (#7), and (#8), there are no derogatory terms or incitement of violence, and the statements do not target specific groups, which suggests that they are not hate speech.\n\nApplying these factors to the given input for (#9), it can be observed that the statement does not contain any derogatory terms or incitement of violence. However, it does target a specific group (feminists) and suggests that men should not help feminists.\n\nBased on the analysis of the provided examples and the content of the given input, it can be concluded that (#9) is a form of hate speech. The statement promotes negative stereotypes and discrimination against feminists and suggests that men should not help feminists, which can contribute to a hostile environment and perpetuate harmful gender stereotypes.

Figure 5: Case Study of ETHOS dataset by Vicuna-13B. The red color refers to the task-specific labels generated by the model. The green color indicates the model's analysis of the test instance. The blue color marks the classification paradigm summarized by the model based on the demonstrations.

1) **Our methods exhibit generalizability across various sizes of LLMs.** As depicted in Table 3, our approaches consistently yield robust performance across different LLMs. Particularly noteworthy is the substantial improvement observed with Vicuna-7B, where our methods surpass the baseline by more than 10 percentage points. Furthermore, our approaches seamlessly integrate with ChatGPT, highlighting the broad applicability and effectiveness of our methods.

2) **Guidelines from Larger LLMs Enhance Smaller Ones, vice versa.** As illustrated in Table 3, integrating insights from larger models significantly enhances the performance of smaller ones, highlighting the invaluable guidance offered by larger LLMs. This enhancement facilitates smaller LLMs in extracting precise semantic modes with greater ease. Conversely, smaller language models might misconstrue semantic modes from demonstrations, leading to the formulation of misleading explanatory directives. Such ambiguity consequently leads to a degradation in performance for larger models.

3) **Larger language models are less susceptible to the influence of misleading guidelines generated by smaller ones.** For certain cases, larger models possess a greater ability to override the erroneous information generated by smaller models in their guidelines, thereby still providing correct answers. For instance, when utilizing guidelines generated by Vicuna-7B, the classification performance of GPT-3.5-turbo is 80.20, surpassing that of LlaMA2-13B-chat, which stands at 73.30.

4.6 Case Study

In Figure 5, we present the results of Vicuna-13B using a test instance from the ETHOS dataset to explore the functionality of our self-explanatory ICL. We employ red, blue, and green colors to highlight the predicted label, the summarized classification paradigms, and instance-level analysis, respectively. We can see that, while vanilla ICL yields inaccurate predictions, our method enhances the model's ability to summarize classification paradigms accurately from demonstrations and generate comprehensive analyses alongside the test instance. This augmentation aids LLMs in precisely extracting semantic modes, thereby leading to more precise predictions.

5 Related Work

5.1 Demonstration Organization

Existing research on in-context learning underscores the impact of diverse demonstration organizations and proposes varied methodologies for optimal demonstration selection and ordering (Zhang et al., 2022a; Gonen et al., 2022; Hu et al., 2022; Poesia et al., 2022; Nie et al., 2023; Scarlatos and Lan, 2023; Li et al., 2023; Xu and Zhang, 2024). In summary, two primary objectives emerge for demonstration selection: similarity-based and diversity-based methods. The former entails choosing demonstrations akin to the test instance, facilitating learning through analogy for LLMs (Liu et al., 2022; Lu et al., 2022; Rubin et al., 2022; Shi et al., 2022; Zhang et al., 2022b; Agrawal et al., 2022; Dalvi Mishra et al., 2022; Yu et al., 2023; Li and Qiu, 2023b; Luo et al., 2023; Wang et al., 2024). The latter emphasizes maximizing demonstration diversity concerning the given test instance to diminish redundancy and enrich information conveyed to LLMs (Sorensen et al., 2022; Levy et al., 2023; Ye et al., 2023; Naik et al., 2023; Ma et al., 2023; Fu et al., 2023). Both methods necessitate selecting and reordering demonstrations for each individual test instance according to the aforementioned selection metrics. We refer to them as *instance-level methods*.

In this paper, we focus on searching for an optimal ordering based on a given set of demonstrations and present our *instance-free* demonstration reordering method. Our method focuses on maximizing the reduction of semantic ambiguity in demonstrations, utilizing label probabilities and entropy in demonstrations as selection objectives, thereby reducing dependency on test instances. In this way, our method significantly reduces costs while enhancing the effectiveness of in-context learning.

5.2 Explanation-Based Methods

Recent research investigates the impact of integrating explanations into inputs to enhance task performance and alleviate feature biases in in-context learning (Ye and Durrett, 2022; Wei et al., 2023a; Si et al., 2023). While these studies rely on human-annotated explanations, our proposed self-explanatory ICL framework removes dependencies on human costs by integrating self-explanations generated by large language models. These self-explanations subsequently assist LLMs in accurately selecting semantic modes.

6 Conclusion

Our study highlights the significant impact of semantic ambiguity within demonstrations on in-context learning, revealing its potential to introduce biases in predictions. To tackle this issue, we propose two tailored de-biasing strategies for in-context learning, named Instance-Free Demonstration Reordering and Self-Explanatory In-Context Learning, which effectively assist LLMs in accurately selecting semantic modes, thereby significantly mitigating demonstration bias. Experiments conducted across six datasets verify the effectiveness of our approaches in substantially mitigating demonstration bias and enhancing the performance of in-context learning. Our findings hold promise for significantly reducing or eliminating the burden on users seeking optimal demonstrations in real-world applications, thus enabling researchers to harness the in-context learning capabilities of large language models more effectively.

7 Limitations

In this paper, we focus on evaluating the effectiveness of our methods in classification tasks using several LLMs. Further research is needed to explore additional tasks and a broader range of LLMs.

Acknowledgements

We sincerely thank all anonymous reviewers for their insightful comments and valuable suggestions. This research work is supported by the National Natural Science Foundation of China under Grants no. 62306303, 62122077 and 62106251. This work is supported by Ant Group Research Fund.

References

Sweta Agrawal, Chunting Zhou, Mike Lewis, Luke Zettlemoyer, and Marjan Ghazvininejad. 2022. In-context examples selection for machine translation. *Preprint*, arXiv:2212.02437.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Bhavana Dalvi Mishra, Oyvind Tafjord, and Peter Clark. 2022. Towards teachable reasoning systems: Using a dynamic memory of user feedback for continual system improvement. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9465–9480, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xiaowen Ding, Bing Liu, and Philip S. Yu. 2008. A holistic lexicon-based approach to opinion mining.

In *Proceedings of the International Conference on Web Search and Web Data Mining, WSDM 2008, Palo Alto, California, USA, February 11-12, 2008*, pages 231–240. ACM.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-based prompting for multi-step reasoning. *Preprint*, arXiv:2210.00720.

Zelalem Gero, Chandan Singh, Hao Cheng, Tristan Naumann, Michel Galley, Jianfeng Gao, and Hoifung Poon. 2023. Self-verification improves few-shot clinical information extraction. *Preprint*, arXiv:2306.00024.

Hila Gonen, Srini Iyer, Terra Blevins, Noah A. Smith, and Luke Zettlemoyer. 2022. Demystifying prompts in language models via perplexity estimation. *Preprint*, arXiv:2212.04037.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. Pre-training to learn in context. *Preprint*, arXiv:2305.09137.

Yushi Hu, Chia-Hsuan Lee, Tianbao Xie, Tao Yu, Noah A. Smith, and Mari Ostendorf. 2022. In-context learning for few-shot dialogue state tracking. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2627–2643, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zixian Huang, Jiaying Zhou, Gengyang Xiao, and Gong Cheng. 2023. Enhancing in-context learning with answer feedback for multi-span question answering. *Preprint*, arXiv:2306.04508.

Brendan King and Jeffrey Flanigan. 2023. Diverse retrieval-augmented in-context learning for dialogue state tracking. *Preprint*, arXiv:2307.01453.

Itay Levy, Ben Bogin, and Jonathan Berant. 2023. Diverse demonstrations improve in-context compositional generalization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1401–1422, Toronto, Canada. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. 2023. Unified demonstration retriever for in-context learning. *Preprint*, arXiv:2305.04320.

Xiaonan Li and Xipeng Qiu. 2023a. Finding support examples for in-context learning. *Preprint*, arXiv:2302.13539.

Xiaonan Li and Xipeng Qiu. 2023b. Mot: Memory-of-thought enables chatgpt to self-improve. *Preprint*, arXiv:2305.05181.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Man Luo, Xin Xu, Zhuyun Dai, Panupong Pasupat, Mehran Kazemi, Chitta Baral, Vaiva Imbrasaite, and Vincent Y Zhao. 2023. Dr.icl: Demonstration-retrieved in-context learning. *Preprint*, arXiv:2305.14128.

Huan Ma, Changqing Zhang, Yatao Bian, Lemao Liu, Zhirui Zhang, Peilin Zhao, Shu Zhang, Huazhu Fu, Qinghua Hu, and Bingzhe Wu. 2023. Fairness-guided few-shot prompting for large language models. *Preprint*, arXiv:2303.13217.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Pekka Malo, Ankur Sinha, Pyry Takala, Pekka Korhonen, and Jyrki Wallenius. 2013. Good debt or bad debt: Detecting semantic orientations in economic texts. *Preprint*, arXiv:1307.5336.

Ioannis Mollas, Zoe Chrysopoulou, Stamatis Karlos, and Grigorios Tsoumakas. 2020. Ethos: a multi-label hate speech detection dataset. *Complex & Intelligent Systems*, pages 1–16.

Ranjita Naik, Varun Chandrasekaran, Mert Yuksekgonul, Hamid Palangi, and Besmira Nushi. 2023. Diversity of thought improves reasoning abilities of large language models. *Preprint*, arXiv:2310.07088.

Ercong Nie, Sheng Liang, Helmut Schmid, and Hinrich Schütze. 2023. Cross-lingual retrieval augmented prompt for low-resource languages. *Preprint*, arXiv:2212.09651.

Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. 2023. What in-context learning "learns" in-context: Disentangling task recognition and task learning. *Preprint*, arXiv:2305.09731.

Gabriel Poesia, Alex Polozov, Vu Le, Ashish Tiwari, Gustavo Soares, Christopher Meek, and Sumit Gulwani. 2022. Synchromesh: Reliable code generation from pre-trained language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671, Seattle, United States. Association for Computational Linguistics.

Alexander Scarlatos and Andrew Lan. 2023. Reticl: Sequential retrieval of in-context examples with reinforcement learning. *Preprint*, arXiv:2305.14502.

Peng Shi, Rui Zhang, He Bai, and Jimmy Lin. 2022. XRICL: Cross-lingual retrieval-augmented in-context learning for cross-lingual text-to-SQL semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5248–5259, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Chenglei Si, Dan Friedman, Nitish Joshi, Shi Feng, Danqi Chen, and He He. 2023. Measuring inductive biases of in-context learning with underspecified demonstrations. *Preprint*, arXiv:2305.13299.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Taylor Sorensen, Joshua Robinson, Christopher Rytting, Alexander Shaw, Kyle Rogers, Alexia Delorey, Mahmoud Khalil, Nancy Fulda, and David Wingate. 2022. An information-theoretic approach to prompt engineering without ground truth labels. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 819–862, Dublin, Ireland. Association for Computational Linguistics.

Xiaofei Sun, Xiaoya Li, Shengyu Zhang, Shuhe Wang, Fei Wu, Jiwei Li, Tianwei Zhang, and Guoyin Wang. 2023. Sentiment analysis through llm negotiations. *Preprint*, arXiv:2311.01876.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Liang Wang, Nan Yang, and Furu Wei. 2024. Learning to retrieve in-context examples for large language models. *Preprint*, arXiv:2307.07164.

Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. 2023. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. *Preprint*, arXiv:2301.11916.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023a. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, and Tengyu Ma. 2023b. Larger language models do in-context learning differently. *Preprint*, arXiv:2303.03846.

Genta Indra Winata, Liang-Kang Huang, Soumya Vadlamannati, and Yash Chandarana. 2023. Multilingual few-shot learning via language model retrieval. *Preprint*, arXiv:2306.10964.

Shangqing Xu and Chao Zhang. 2024. Misconfidence-based demonstration selection for llm in-context learning. *Preprint*, arXiv:2401.06301.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. *Preprint*, arXiv:2309.03409.

Jiacheng Ye, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2023. Compositional exemplars for in-context learning. *Preprint*, arXiv:2302.05698.

Xi Ye and Greg Durrett. 2022. The unreliability of explanations in few-shot prompting for textual reasoning. *Preprint*, arXiv:2205.03401.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. 2023. Generate rather than retrieve: Large language models are strong context generators. *Preprint*, arXiv:2209.10063.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022a. Active example selection for in-context learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9134–9148, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022b. Automatic chain of thought prompting in large language models. *Preprint*, arXiv:2210.03493.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

A Instruction Optimization

The mismatch between input and output in our self-explanatory in-context learning framework introduce challenges in crafting instructions. In order to find optimal instructions and make the model's generated outputs more controllable, we employ a straightforward yet effective instruction optimization method called Optimization by PROmpting (OPRO, Yang et al. (2023)). In this method, the optimization task is described in natural language and large language models are utilized as optimizer. As shown in Figure 6, the input comprises three key components:

1. **Task Description**: Derived from natural language, the task description guides the *Optimizer LLM* in comprehending and iteratively refining instructions based on pre-existing ones.

2. **Optimization Trajectory**: Pre-existing instructions and their corresponding performance scores on the training set by the *Scorer Model*. It serves as an alternative loss function, aiding large language models in understanding the relative effectiveness of various instructions. It's important to note that the Scorer Model may differ from the Optimizer LLM.

3. **Task Examples**: A limited set of input-output examples from real-world application tasks is provided to enhance the Optimizer LLM's understanding of the practical application of the given instructions.

In our paper, we manually designed three instructions as seed instructions and utilized SST-2 dataset as our training set. We use GPT-3.5-turbo as the optimizer LLM and the Vicuna-13B as the scorer LLM respectively. Throughout the optimization process, we set the sampling temperature to 1.0. The primary objective was to enable the optimizer model to effectively leverage previously identified instructions while also navigating away from local optima to discover a broader range of instructions. Given the volatility in the model's generation outcomes, we concurrently generated 8 new instructions at each optimization step to enhance overall stability.

To prioritize the emphasis on superior instructions, we retained the top-performing 20 instructions in the optimization trajectory. Furthermore,

User:
Below are some texts along with their corresponding scores. The texts are arranged in ascending order based on their scores, where higher scores indicate better quality.

text: Give me the output and explanation of (#5). score: 30.0
(... More instructions and scores ...)

The following example show how to apply your text: you replace <INS> in each input with your text, then read the input and give an output.

input:
<INS>
(#1) Input: appropriately Output: Positive
...
(#5) Input: it makes your [...] Output:
output:
Negative Sentiment
(... More examples ...)

Write your new text that is different from the old ones and has a score as high as possible.

Optimizer LLM:
Give me the explanation and result of (#33) based on the previously provided examples.
Score: 50.0

Figure 6: An example of the input of OPRO, where the generated instruction will be prepended to the beginning of demonstrations. <INS> denotes the position where the generated instruction will be added. The red text describes the optimization task and output format; the blue text contains solution-score pairs; the purple text are task descriptions.

we incorporated a strategy of randomly sampling 4 test examples from the test set into the task description at each optimization step. Each test example comprised 4 unique demonstrations, in order to mitigate the optimizer model's tendency to overly concentrate on specific instances, fostering a broader global perspective, thereby improve its overall performance and robustness. Ultimately, we performed 50 steps of and chose the instruction that demonstrated the best performance on the development set as our self-explanatory instruction.