PACIT: Unlocking the Power of Examples for Better In-Context Instruction Tuning

Anonymous ACL submission

Abstract

Instruction tuning enhances the instruction following ability of large language models by finetuning with supervised instruction data. Previous work proposes in-context instruction tuning (ICIT) where specific positive or negative examples are incorporated into the prompt for better performance. In this work, we propose PACIT, a simple and effective in-context instruction tuning method, inspired by the pedagogical concept of *desirable difficulty*. The PACIT method unlocks the power of examples by encouraging the model to actively learn to grasp the distinctions between the positive and negative examples instead of merely reading. The model is expected to first verify the correctness of the provided example according to the task description, which is then set as the condition for generating a better response to the task instance. Our extensive experiments prove the effectiveness of PACIT, outperforming ICIT baseline on both in-domain and out-domain tasks up to 9.16 and 3.14 average ROUGE-L scores, respectively. Moreover, PACIT can notably enhance the performance of instruction tuning even when all positive and negative examples are generated with a self-instruct method.

1 Introduction

Large language models (LLMs) have garnered significant interest from both academia and industry due to their superior performance on a variety of natural language processing tasks such as question answering and text generation. Instruction tuning (IT; Ouyang et al. 2022) optimizes the pretrained language models with supervised instruction data to enhance the capabilities of the instruction following and zero-shot generalization to unseen tasks (Chung et al., 2022; Ouyang et al., 2022; Sanh et al., 2022; Taori et al., 2023; Xue et al., 2023). InstructGPT (Ouyang et al., 2022) proposes in-context instruction tuning (ICIL) where the LLM is finetuned using instruction data with few-shot human-crafted positive examples. SuperNI (Wang et al., 2022) presents a variant of in-context instruction tuning by further incorporating specified positive and negative examples in each task. The ICIL method achieves significant improvement compared with the vanilla zero-shot instruction tuning method (Ouyang et al., 2022; Wang et al., 2022; Li et al., 2023a) with the knowledge from the demonstrations.

However, previous in-context instruction tuning merely shows the specified positive and negative examples in the prompt, without further considerations for better digestion of examples. LLMs still struggle to follow the instructions precisely in some scenarios (Li et al., 2023b; AlShikh et al., 2023), which hinders their further applications.

In this work, we introduce PACIT, a simple and novel in-context instruction tuning approach (see Figure 1) inspired by the pedagogical concept of *desirable difficulty* (Wikipedia, 2023; Marsh and Butler, 2013). During finetuning with PACIT method, the model first accomplishes a quiz about the judgment of correctness of the provided examples based on the task description, then responds to the task instance input. By transforming the provided example into a related quiz of the simple classification task, we encourage the model to be actively involved in recalling correlated information and grasping the distinction between positive and negative examples, going beyond surface-level information. In contrast to simply reading the examples, this approach enhances the model's comprehension of the task information, thereby improving its ability to follow instructions.

Extensive experiments prove the effectiveness of PACIT, outperforming ICIT baseline up to 9.16 and 3.14 average ROUGE-L (Lin, 2004) on in-domain and out-of-domain datasets of SuperNI (Wang et al., 2022), respectively. The PACIT still consistently surpasses traditional methods when the positive and negative examples are synthesized with self-instruct (Wang et al., 2023) by ChatGPT (OpenAI, 2022). Therefore, in cases that the human-crafted positive and negative examples are not available, the PACIT has the potential to be a better instruction tuning strategy even for a large-scale instruction dataset. Our contributions are summarized as follows:

- We propose PACIT, a simple yet effective in-context instruction tuning method that achieves better instruction following ability by better grasping the differences between positive and negative examples.
- Extensive experiments demonstrate the superior performance of PACIT over competitive baselines consistently across in-domain and out-domain datasets.
- The PACIT also achieves better performance than vanilla instruction tuning when the examples are all synthesized with the self-instruct method.[1]

2 Related Work

2.1 Instruction Tuning

Instruction tuning (Ouyang et al., 2022) finetunes the pretrained language models with supervised instruction data to enhance the instruction following ability and enable the zero-shot generalization to unseen tasks (Chung et al., 2022; Wei et al., 2022; Ouyang et al., 2022; Sanh et al., 2022; Taori et al., 2023). The instruction tuning is an essential training stage for most large language models (Ouyang et al., 2022; Taori et al., 2023). It commonly uses the next token prediction as the training objective.

The key to instruction tuning is the quality and diversity of the instruction data (Zhou et al., 2023). The instruction data used by InstructGPT (Ouyang et al., 2022) is created with human experts. It can also be created with LLMs like ChatGPT (OpenAI, 2022) with self-instruct (Wang et al., 2023) method. The self-instruct method synthesizes instruction data by prompting the LLM with few-shot examples and guidelines to use instructional signals from the model itself for data augmentation. The evol-instruct (Xu et al., 2023) method further improves self-instruct to create more diverse instruction data with varying levels of complexities. The humpback (Li et al., 2023c) proposes to iterativly optimize the model and generate high-quality instruction data without the reliance on strong proprietary LLMs, similar to the back-translation practice in machine translation. Super natural instructions (SuperNI; Wang et al. 2022) is a benchmark that covers 76 distinct task types of 1616 diverse NLP tasks, including but not limited to classification, extraction, infilling, sequence tagging, text rewriting, and text composition. Each task in the SuperNI benchmark contains the task definition, task instances and example instances. Both task instance and example instance contain the input-output pairs for the task. The example instances have additional tags (i.e., positive or negative) based on the example and the task description.

In-context instruction tuning (Ouyang et al., 2022; Wang et al., 2022; Li et al., 2023a) finetune the LLMs with supervised instruction data as well as task-specific examples. The few-shot examples used in InstructGPT are all human-crafted positive examples. Wang et al. (2022) further incorporates specified positive and negative crafted examples into the in-context instruction tuning. Li et al. (2023a) explore the in-context instruction tuning in the multimodal domain. Different from previous works that simply have the model passively read the examples, we explore to encourage the model to actively learn about the examples via verification the correctness of examples.

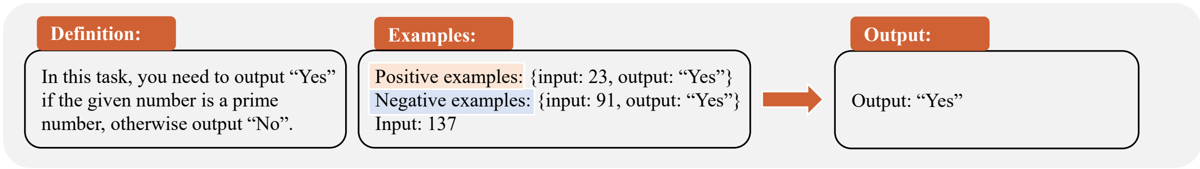
2.2 In-Context Learning

In-context learning (ICL; Liu et al. 2022; Rubin et al. 2022; Min et al. 2022a) is a prompt-based method that encourages the language models to learn from the few-shot examples presented in the model input. Researchers explore different approaches to improve the performance of ICL. Min et al. (2022a) and Chen et al. (2022) introduce meta-learning to better adapt the language models to ICL. Zhao et al. (2021) estimates models' bias towards each answer and then develop contextual calibration to adjust the model's output probabilities. SG-ICL (Kim et al., 2022) proposes to generate demonstration examples for in-context learning from the language model itself instead of humans. Active Prompting (Diao et al., 2023) selects the most uncertain questions as demonstration examples to further improve the performance. Min et al. (2022b) finds that replacing gold labels with random labels only marginally hurts performance, which indicates models learn from the example format rather than input-label pairs. Yoo et al. (2022) revisit previous findings of Min et al. (2022b) and introduce novel metrics to prove that the input-label correspondence plays a more significant role in contextual demonstration than previously considered. However, most of these methods focus on the inference stage and explicitly show the correctness of the demonstration examples. Our work focuses on the instruction tuning stage.

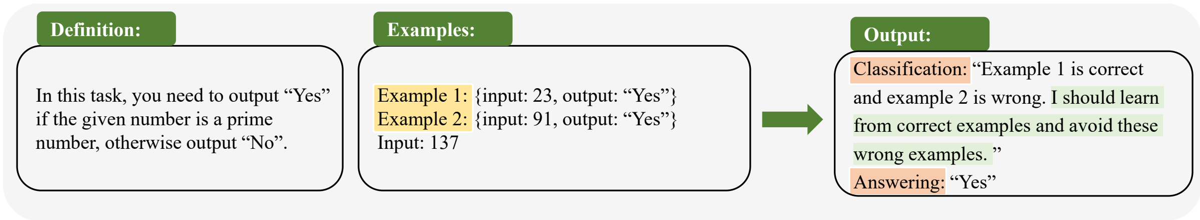
3 Method

In this work, we focus on the in-context instruction tuning (Wang et al., 2022) where both positive and negative examples are provided as the case in the SuperNI dataset (see Figure 1). The model is trained to generate a response that is similar to the positive examples while avoiding the mistakes in the negative ones. Conventional works merely present these examples and their tags in the prompt following the practice of in-context learn-

[1] Our code and models will be made public.



(a) Vanilla In-context Instruction Tuning (SuperNI (Few-Shot))



(b) Our proposed PACIT approach

Figure 1: The overview of PACIT. PACIT consists of two stages: Classification and Answering. (1) **Classification:** Judge the correctness of each provided example based on the task description and then take the self-reminder action. (2) **Answering:** Respond to the main task instruction conditioned on the classification results. Two stages are executed sequentially within a single data sample.

ing. We propose PACIT for better in-context instruction tuning by unlocking the power of provided examples. The PACIT is motivated by the pedagogical psychological concept of *desirable difficulty* (Marsh and Butler, 2013; Wikipedia, 2023), which improves the long-term performance of students by a learning task that requires a considerable but desirable amount of effort.

As an example of desirable difficulty, quizzing oneself with flashcards brings better learning outcomes than just reading the materials, as the quizzes require students to consistently recall associated information and encourage them to learn the material more concretely and actively. Simply reading the materials results in lower engagement and less attention from students. The key information and connected knowledge of the materials may be overlooked. In contrast, students think, analyze and try to apply their existing knowledge when they tackle a problem by hand. Active involvement in learning enhances their understanding of the knowledge, leading to better learning outcomes.

Following the insight of *desirable difficulty*, the PACIT proposes a supplementary quiz with the examples and asks the model to first accomplish the quiz before the task mentioned in the instruction. As shown in Figure 1, the model is required to first classify the examples presented in the prompt into two types, positive or negative, according to the task description. The negative example indicates the unsatisfied output for the given input for this task, which should be avoided. After that, the model generates the response to the instruction based on the classification result of the provided examples. In this way, the model actively learns about the examples by accomplishing the related quiz, which further facilitates the understanding and grasp of the given task.

Consistent with SuperNI, each task has a task description S_T, a training dataset $\mathcal{D} = \{(X, Y)\}$, and an example pool consisting of positive and negative examples. For each input-output instance pair (X, Y) in $\mathcal{D}$, we randomly select k examples from the example pool and determine the order of positive and negative examples randomly. Both the input and output of examples are presented in the prompt ($S_e^{in} = \{X_e, Y_e\}$), while the corresponding label L_e (i.e., positive or negative) is set as the answer to the supplementary quiz and is part of the model output (see the example in Figure 1). The ground-truth label of each example is replaced with the ordinal number and concealed in the input. In this way, the supplementary quiz is designed without human effort. Each data sample in PACIT has two stages, i.e. **Classification** and **Answering**.

Classification The model is expected to judge the correctness of each provided example based on the task description during the classification stage. The ground-truth classification result J_e is created from a template shown in Figure 1 and the example tag L_e. After giving the answer to the quiz, the model continues to generate the corresponding action to be taken A_e (e.g., "I should learn from correct examples and avoid mistakes in the wrong examples."). The action serves as a self-reminder to encourage the model to take the corresponding action for better performance. During the first classification stage, the model is optimized with the next token prediction training objective. The ground-truth for action A_e are human-crafted without tuning and kept the same for all samples. All tokens in the classification result and action are counted for the loss calculation. Formally, the loss of the classification stage can be represented as:

$$\mathcal{L}_{\mathbf{c}} = -\sum_{(X,Y)\in\mathcal{D}} \log P(J_e, A_e | S_T, S_e^{in}, X; \theta). \quad (1)$$

Answering Based on the result of the supplementary quiz J_e and the corresponding action A_e, the model is elicited to output the answer Y for instance input X in the task. The answering stage is also trained with the language modeling objective. The corresponding training loss is calculated as

$$\mathcal{L}_{\mathbf{a}} = -\sum_{(X,Y)\in\mathcal{D}} \log P(Y | S_T, S_e^{in}, X, J_e, A_e; \theta). \quad (2)$$

The overall training loss of PACIT is the sum of these two losses $\mathcal{L} = \mathcal{L}_{\mathbf{c}} + \mathcal{L}_{\mathbf{a}}$. During inference, the model generates the answer in the main task after completion of the auxiliary classification task.

4 Experiments

4.1 Experiment Setting

Dataset We conduct experiments on the SuperNI-V2 dataset (Wang et al., 2022), an open-source dataset comprising over 800+ English tasks with diverse task types. Each task in the dataset includes four components: task definition, positive examples, negative examples and explanations. To ensure consistency, we utilize the same dataset split as SuperNI: the training set consisting of 756 diverse tasks and a hold-out test set containing 119 unseen out-domain tasks for evaluation purposes. Additionally, we construct a held-in test set that mirrors the training set's tasks but with different task instances to prevent any data leakage. As the performance saturates when the number of instances per task increases (Wang et al., 2022), we randomly sample 60 instances for each task in the training set. For the test set, we randomly sample 100 instances for each task of the held-out test set and 15 instances for each task of the held-in test set, ensuring a comparable total number of instances for both datasets. The statistics of our training, held-in and held-out datasets are presented in Table 1.

Statistics	Train Set	Held-In	Held-Out
Number of tasks	756	756	119
# of total instances	45360	11340	11900
Avg. # of Ex.	1.83	1.79	1.75

Table 1: Statistics of our training, held-in, and held-out datasets. 'Avg. # of Ex.' denotes the average number of examples per task.

Construction of Dataset. To perform in-context instruction tuning, we construct the training dataset with data samples of the format *task definition+positive/negative examples+task instance*. For each data sample, examples are added incrementally until the maximum input length is reached. Specifically, given a task instance, we first include the *instance* and its corresponding *task definition* to form a data sample. Subsequently, we randomly select a positive example and a negative example for the task and gradually add them to the data sample. To prevent the model from simply memorizing the corresponding tags, the order of the examples is shuffled. If adding an example exceeds the maximum input length limit, the addition process is stopped. This process results in four distinct types of data samples: (1) **Without examples**: training samples without any examples. (2) **Only positive example**: training samples with only one positive example. (3) **Only negative example**: training samples with only one negative example. (4) **Mixing examples**: training samples with both positive and negative examples. The proportions of these four types within our training data are 2.9%, 6.3%, 0.5% and 90.2%, respectively. The few-shot inference dataset is constructed similarly, while the zero-shot inference dataset consists of data samples with the format *task definition+task instance*.

Settings and Metrics Following Kung and Peng (2023), we utilize two variants of `T5-LM-Adapt` (Raffel et al., 2020) as the backbones of PACIT: `T5-Large-lm-adapt-770M` (T5-770M) and `T5-XL-lm-adapt-3B` (T5-3B). Additionally, to evaluate PACIT with a stronger backbone, we conduct experiments using the `LLaMA-2-7B` (LLaMA2-7B) model. During inference, we employ greedy decoding (i.e., set the temperature to 0) following Wang et al. (2022) to obtain the most confident predictions from the model outputs. Given the diversity of tasks and the open-ended generation nature of formulation, we adopt ROUGE-L metric (Lin, 2004) for reporting aggregated performance results. The metric has been shown to correlate well with accuracy for classification tasks and human evaluation (Wang et al., 2022). Unless otherwise specified, we report results on the held-out dataset in the Ablation Study (Section 4.3) and Analyses (Section 5).

Training Details We use Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ to finetune the models. The models are trained for five epochs and the last checkpoint is used for evaluation. The global batch size is 64. We use the linear learning rate scheduler. The learning rate for T5-based models is set to 2×10^{-4} following Kung and Peng (2023), while the learning rate for LLaMA-2 is set to 2×10^{-5} following Taori et al. (2023); Chen et al. (2023b). We set the maximum input length as 1024 and

Model	Testing Setting → Training Setting ↓	Held-Out			Held-In		
		Zero-Shot	Few-Shot	Avg ROUGE-L	Zero-Shot	Few-Shot	Avg ROUGE-L
`T5-770M`	SuperNI (Zero-Shot)	**38.02**	40.59	39.30	**46.22**	42.59	44.40
	SuperNI (Few-Shot)	33.30	45.08	39.19	43.59	52.96	48.27
	PACIT	33.59	**46.66**	**40.13**	44.67	**53.31**	**48.99**
`T5-3B`	SuperNI (Zero-Shot)	42.89	45.73	44.31	**49.95**	47.59	48.77
	SuperNI (Few-Shot)	38.54	51.08	44.81	41.49	52.96	47.23
	PACIT	**43.09**	**52.11**	**47.60**	47.29	**55.21**	**51.25**
`LLaMA2-7B`	SuperNI (Zero-Shot)	44.81	49.35	47.08	49.36	48.85	49.10
	SuperNI(Few-Shot)	42.14	50.71	46.43	45.53	52.68	49.10
	PACIT	**45.62**	**53.53**	**49.57**	**54.05**	**62.47**	**58.26**

Table 2: The comparision results of PACIT and baselines under zero-shot and few-shot inference settings on hold-in and hold-out datasets. **Avg ROUGE-L:** we calculate the averaged ROUGE-L under zero-shot and few-shot inference settings. **Bold** denotes the best result.

the maximum output length as 128 for all models following Wang et al. (2022). All experiments are run on eight NVIDIA RTX-4090 GPUs using `Huggingface Transformers`[2] toolkit.

Baselines We compare PACIT with two baselines:

- SuperNI (Zero-Shot): We formulate each data sample as *task definition+main task instance* and train with conventionally instruction tuning method. No examples are used during training for this setup.
- SuperNI (Few-Shot): We use the same training dataset as PACIT, but train with conventionally in-context instruction tuning. In the subsequent text, we may use SuperNI to denote this method for simplicity.

4.2 Main Results

To assess the efficacy of PACIT, we compare it with baselines as presented in Table 2. As can be seen, PACIT consistently outperforms SuperNI (Zero-Shot) and SuperNI (Few-Shot) methods across the held-in and held-out datasets. Notably, the performance gap is more pronounced for larger models compared to smaller model. Specifically, when utilizing the T5-3B and LLaMa2-7B models, PACIT exhibits substantial improvements over the SuperNI (Few-Shot) method, with average ROUGE-L score boosts of 2.79 and 3.14 on the held-out test set, and 4.02 and 9.16 on the held-in test set, respectively. Conversely, smaller T5-770M model demonstrates only marginal increases of 0.94 and 0.72 average ROUGE-L scores. We hypothesize that larger models, which have stronger learning capabilities, can excavate more internal information in demonstration examples with our proposed PACIT methods. Additionally, it is noteworthy that PACIT exhibits greater improvements on the held-in datasets compared to the held-out datasets, indicating its ability to significantly benefit seen tasks. In the zero-shot inference setting, SuperNI (Zero-Shot) method achieves good performance. However, its performance sharply declines in the few-shot setting. This discrepancy can be attributed to the importance of maintaining consistency between the training and inference settings In summary, PACIT outperforms all baselines and achieves new state-of-the-art on ICIT.

[2]`https://github.com/huggingface/transformers`

ID	Method	ZS	FS	Avg.
(1)	PACIT	**43.09**	**52.11**	**47.60**
(2)	(1)−action	41.48	51.29	46.38
(3)	(2)−aux.	38.50	51.08	44.81

Table 3: The performance (ROUGE-L) of ablation study variants (ZS=zero-shot inference, FS=few-shot inference) on held-out set. Starting from PACIT, we gradually remove the action (ID=2) and the auxiliary classification stage (aux., ID=3) in each data sample.

4.3 Ablation Study

We conduct an ablation study on the training method of PACIT. Initially, we begin with PACIT, which consists of two training stages: classification with action, and answering. Subsequently, we gradually remove the action after classification (setting (2)) and the whole classification stage to roll back to the vanilla SuperNI (Few-Shot) method (setting (3)).

The results are shown in Table 3. Removing the action leads to a decrease of 1.22 average ROUGE-L score, and further removing the classification stage results in an additional decrease of 1.57 average ROUGE-L score. This observation confirms

Definition : Two analogies that relate items to the associated containers is given in the form " A : B . C : ?" . " A : B " relates item A to its associated container B . Your task is to replace the question mark (?) with the appropriate container for the given item C , following the " A : B " relation . Positive Example 1 - Input : jam : jar . cereal : ? Output : box . Negative Example 1 - In put : detergent : bottle . cereal : ? Output : cupboard . Now complete the following example - Input : money : wallet . milk : ? Output : container ✗

(a) SuperNI (Few-Shot)

Definition : Two analogies that relate items to the associated containers is given in the form " A : B . C : ?" . " A : B " relates item A to its associated container B . Your task is to replace the question mark (?) with the appropriate container for the given item C , following the " A : B " relation . Example 1 - Input : jam : jar . cereal : ? Output : box . Example 2 - Input : detergent : bottle . cereal : ? Output : cupboard . Now complete the following example - Input : money : wallet . milk : ? Output : bottle ✓

(b) PACIT

Figure 2: A concrete example of attention visualization for SuperNI (Few-Shot) and PACIT methods.

our insights regarding *desirable difficulty*, as the inclusion of a supplementary quiz on the examples and an action to emphasize its importance guides the model to enhance its learning from the examples.

5 Analyses

The Visualization of Attention. To better understand how PACIT works, we conduct a case study by visualizing the attention weights in T5-3B model. We visualize the averaged encoder-decoder attention weights of different heads in the last layer of T5-3B. Figure 2 shows a concrete example of PACIT v.s. SuperNI (Few-Shot). The color in each figure represents the relative attention weights. As can be seen, PACIT allocates more attention to the task definition and examples' information compared with the SuperNI (Few-Shot) model. The attention weights from PACIT exhibit a broader span across the prompt. This observation is expected as the classification task in PACIT encourages the model to focus more on task definition and examples, otherwise it cannot classify examples correctly. We also manually check some other examples which present similar patterns.

The Relationship between Classification Accuracy and Model Performance. To gain insights into the correlation between the auxiliary task (i.e., classification) and main task, we analyze the training dynamics by plotting the main task's performance (ROUGE-L) against the auxiliary task's performance (Acc). The results are shown in Figure 3. The classification accuracy demonstrates a strong correlation with the main task's ROUGE-L score, as evidenced by the slope. Furthermore, we calculate the Pearson correlation coefficient between these two metrics, resulting in a high value of 0.98. While correlation does not establish causation, it does provide valuable insights into the interpretability of PACIT.

The Effect of Classification Labels in Training and Inference Phase. Inspired by previous work on in-context learning (Min et al., 2022b; Madaan et al., 2023; Wei et al., 2023), we suspect PACIT utilize examples either by (a) recognizing the task from examples and applying LLMs' pretrained priors (learning the format (Min et al., 2022b)) and/or (b) learn the input–label mappings from the presented examples (learning the input-label mapping). When ground-truth labels are provided during in-context instruction tuning, these two factors operate simultaneously. To study which of these factors drives performance, we compare two training settings:

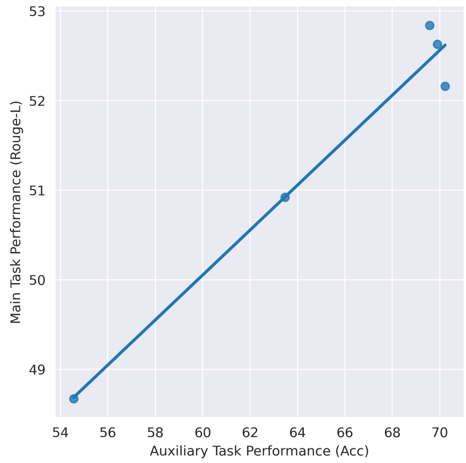


Figure 3: The training dynamics of the main task (ROUGE-L) v.s. the auxiliary classification task (Acc). **Acc:** The accuracy of classification. **ROUGE-L:** The performance of main tasks. The five data points represent five checkpoints obtained after each epoch.

- **Ground-Truth:** The true classification labels are used, which is the standard setup of PACIT.
- **Random:** The classification labels are uniformly sampled from the label space. In this setup, LLMs can only learn the format.

Table 4 shows the results. At the inference stage, in addition to the standard inference setup of PACIT that generates classification labels from the model (**Generated**), we also explore Ground-Truth and Random variants. As can be seen, PACIT with Ground-Truth training setting exhibits a significantly greater improvement over Random training setting on large model (T5-3B) compared to small model (T5-770M). This observation shares some commonalities with previous research on in-context learning, which suggests that **learning**

Model	Testing Setting → / Training Setting ↓	Zero-Shot	Few-Shot		
			Generated	Ground-Truth	Random
T5-770M	SuperNI (Ground-Truth)	33.30	-	45.08	45.26
	SuperNI (Random)	30.66	-	43.54	43.48
	PACIT (Ground-Truth)	33.58	**46.66**	**46.67**	**46.72**
	PACIT (Random)	**34.23**	46.17	46.10	46.11
T5-3B	SuperNI (Ground-Truth)	38.54	-	51.08	51.25
	SuperNI (Random)	36.71	-	49.12	48.92
	PACIT (Ground-Truth)	**43.09**	**52.11**	**52.17**	**52.07**
	PACIT (Random)	33.52	45.76	46.14	46.11

Table 4: The Performance (ROUGE-L) on held-out set with different classification labels in the training and inference time. We compare two training settings and three inference settings for the labels of few-shot examples in each data sample. **Generated**: classification labels generated from the model; **Ground-Truth**: true classification labels; **Random**: randomly sampled classification labels.

Model	Testing Setting → / Training Setting ↓	Zero-Shot	Few-Shot	Avg. ROUGE-L
T5-770M	SuperNI (1 pos and 1 neg)	33.30	45.08	39.19
	SuperNI (2 pos and 2 neg)	30.75	45.82	38.28
	PACIT (1 pos and 1 neg)	**33.59**	**46.66**	**40.13**
	PACIT (2 pos and 2 neg)	28.66	45.85	37.26
T5-3B	SuperNI (1 pos and 1 neg)	38.54	51.08	44.81
	SuperNI (2 pos and 2 neg)	35.72	49.64	42.68
	PACIT (1 pos and 1 neg)	**43.09**	**52.11**	**47.60**
	PACIT (2 pos and 2 neg)	38.92	51.41	45.17

Table 5: The performance (ROUGE-L) on held-out set with different numbers of demonstration examples in zero-shot and few-shot inference settings. **N pos and M neg:** There are N positive examples and M negative examples in each training sample at most.

the format is a broader capability across scales, while learning the input-label mapping is enabled with scale (Wei et al., 2023; Pan et al., 2023; Kossen et al., 2023). We speculate that large model is better at learning input-output mapping than small model. When comparing different inference setups, we find that the model tuned by PACIT is insensitive to labels at the inference stage for both small and large models. This aligns with previous work's (Wei et al., 2023) observation that instruction-tuned models are more reliable on their own semantic priors so that they are less influenced by the labels presented in examples of ICL. All of the aforementioned observations similarly apply to the SuperNI method, suggesting that ICIT shares similarities with in-context learning. We leave more in-depth studies as future work.

The Influence of Number of Demonstration Examples. Humans can improve their ability to complete downstream tasks by learning from more demonstration examples. Therefore, we construct experiments to explore whether more examples in each data sample lead to better performance. The results are shown in Table 5. We use the same number of demonstration examples in both training and few-shot inference time. Overall, more examples consistently lead to performance degradation for both SuperNI and PACIT in zero-shot and few-shot settings. For example, the performance of PACIT on T5-770M and T5-3B drops by 2.86 and 2.43 average ROUGE-L when switching from a pair of positive and negative examples to two pairs, respectively. We suspect with more demonstration examples, PACIT as well as SuperNI could be misguided by interference among examples and their spurious correlations. A similar phenomenon has been observed in in-context learning. We refer the readers to Chen et al. (2023a) for more detailed discussions.

Model	Testing Setting → Training Setting ↓	Zero-Shot	Few-Shot	Avg ROUGE-L
T5-770M	SuperNI (Zero-Shot)	**32.66**	37.50	35.08
	SuperNI (Few-Shot)	23.08	40.54	31.81
	PACIT	32.62	**41.16**	**36.89**
T5-3B	SuperNI (Zero-Shot)	37.63	41.53	39.58
	SuperNI (Few-Shot)	36.38	43.09	39.73
	PACIT	**37.95**	**44.23**	**41.09**

Table 6: The Performance (ROUGE-L) with generated examples (by Self-Instruct) in zero-shot and few-shot inference settings.

The Performance of PACIT with Generated Examples. A limitation of PACIT is its reliance on positive and negative examples during training. However, the positive and negative examples are not readily available for many instruction datasets. As human annotation is expensive and time-consuming, we tackle the problem by leveraging automatically generated examples from LLM. Specifically, we generate examples with the self-instruct (Wang et al., 2023) method, which is a framework for improving the instruction-following capabilities of LLMs by bootstrapping off their own generations. We choose the ChatGPT (`gpt-3.5-turbo-0613`) as the backbone LLM and set the temperature to 0.7 to improve the diversity of generated data. To create our example seed pool, we randomly select eight pairs of positive and negative examples in total from all examples of different tasks. For each generation, we construct the prompt with task definition and few-shot demonstrations to generate new pairs of positive and negative examples. The few-shot demonstrations consist of four pairs of positive and negative examples and their corresponding task definitions randomly sampled from the seed pool. In this way, we reduce the number of annotated training examples from 1384 to 8. Due to the API expense of the proprietary LLM, we only construct 5040 training samples (84 different tasks with 60 training samples each). The entire data template for generating new positive and negative examples is shown in the appendix A (see Figure 5).

The performance with generated examples is shown in Table 6. As can be seen, with generated examples, PACIT improves over baseline without any examples (SuperNI (Zero-Shot)) by 1.81 Avg ROUGE-L on T5-770M and 1.51 Avg ROUGE-L on T5-3B, and vanilla in-context instruction tuning baseline (SuperNI (Few-Shot)) by 5.08 Avg ROUGE-L on T5-770M and 1.63 Avg ROUGE-L on T5-3B. These results are particularly impressive considering that the quantity of our samples accounts for only 11% of the samples used in the main experiment and the generated examples from self-instruct are noisy (Wang et al., 2023). Furthermore, we find that the improvement brought by PACIT over SuperNI (Zero-Shot) is larger for T5-770B compared with T5-3B. This finding contrasts with the main experiments, where T5-3B exhibits an additional 2.46 average ROUGE-L improvement over T5-770M. This disparity can be attributed to small model's limited ability to learn from the input-label mapping, as its performance is less affected by noisy labels generated by self-instruct.

6 Conclusions

In this paper, we introduce PACIT, an effective in-context instruction tuning approach that unlocks the power of examples to enhance the instruction following ability of LLMs. Inspired by the pedagogical observations, PACIT proposes to encourage the model to actively learn and comprehend the differences between the provided positive and negative examples rather than passively reading them. The model completes a quiz to assess the correctness of examples first and subsequently responds to the main task instruction based on the grasp of the examples. Experiments on SuperNI dataset demonstrate the superior performance of PACIT over competitive baselines. In our preliminary experiment, PACIT is observed to improve the performance of instruction tuning with positive and negative examples created with the self-instruct method, which shows a promising approach for better instruction tuning with large-scale instruction data. However, the generated examples with self-instruct method need further filtering to enhance the performance of PACIT as the noisy examples may have negative impact on the performance. We leave the exploration of filtering the augmented data as well as scaling PACIT to larger models like LLaMA-2-13B, LLaMA-2-70B and larger datasets as future work.

Limitations

Compared with the vanilla instruction tuning method without any example, the PACIT achieves better instruction following performance but has higher computation cost as both the input and output have more tokens. During inference, the computation overhead brought by the input examples can be mitigated with efficient inference techniques for long context scenarios such as KV caching (Kwon et al., 2023). For a given task, the representations of examples are computed once and cached in the memory for future use, thus avoiding the recomputation of the examples for each instance. In addition, the proposed PACIT method requires both positive and negative examples which are not readily available for many instruction datasets. These examples can be created with human efforts, resulting in additional expenses. They can also be synthesized with self-instruct method or other LLM-based data augmentation methods. In this case, the generated data samples need to undergo additional filtering following the common practice of data augmentation.

References

Waseem AlShikh, Manhal Daaboul, Kirk Goddard, Brock Imel, Kiran Kamble, Parikshith Kulkarni, and Melisa Russak. 2023. Becoming self-instruct: introducing early stopping criteria for minimal instruct tuning.

Jiuhai Chen, Lichang Chen, Chen Zhu, and Tianyi Zhou. 2023a. How many demonstrations do you need for in-context learning?

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2023b. Alpagasus: Training a better alpaca with fewer data.

Yanda Chen, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. 2022. Meta-learning via language model in-context tuning.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, and et.al. 2022. Scaling instruction-finetuned language models.

Shizhe Diao, Pengcheng Wang, Yong Lin, and Tong Zhang. 2023. Active prompting with chain-of-thought for large language models.

Hyuhng Joon Kim, Hyunsoo Cho, Junyeob Kim, Taeuk Kim, Kang Min Yoo, and Sang goo Lee. 2022. Self-generated in-context learning: Leveraging auto-regressive language models as a demonstration generator.

Jannik Kossen, Yarin Gal, and Tom Rainforth. 2023. In-context learning learns label relationships but is not conventional learning.

Po-Nien Kung and Nanyun Peng. 2023. Do models really learn to follow instructions? an empirical study of instruction tuning.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA. Association for Computing Machinery.

Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Fanyi Pu, Jingkang Yang, Chunyuan Li, and Ziwei Liu. 2023a. Mimic-it: Multi-modal in-context instruction tuning.

Shiyang Li, Jun Yan, Hai Wang, Zheng Tang, Xiang Ren, Vijay Srinivasan, and Hongxia Jin. 2023b. Instruction-following evaluation through verbalizer manipulation.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. 2023c. HumpBack: Self-Alignment with Instruction Backtranslation. ArXiv:2308.06259.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

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 The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online.

Aman Madaan, Katherine Hermann, and Amir Yazdanbakhsh. 2023. What makes chain-of-thought prompting effective? a counterfactual study. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1448–1535, Singapore. Association for Computational Linguistics.

Elizabeth J. Marsh and Andrew C. Butler. 2013. Memory in educational settings. In *The Oxford Handbook of Cognitive Psychology.*, Oxford Library of Psychology., pages 299–317. Oxford University Press, New York, NY, US.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022a. MetaICL: Learning to learn in context. In *Proceedings of the*

2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2791–2809, Seattle, United States.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022b. Rethinking the role of demonstrations: What makes in-context learning work?

OpenAI. 2022. Introducing chatgpt. https://openai.com/blog/chatgpt.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. 2023. What in-context learning "learns" in-context: Disentangling task recognition and task learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8298–8319, Toronto, Canada. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, and et.al. 2022. Multitask prompted training enables zero-shot task generalization.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, and et.al. 2022. Super-Natural Instructions: Generalization via declarative instructions on 1600+ nlp tasks.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners.

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

Wikipedia. 2023. Desirable difficulty — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Desirable_difficulty&oldid=1187704916.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions.

Tianci Xue, Ziqi Wang, and Heng Ji. 2023. Parameter-efficient tuning helps language model alignment.

Kang Min Yoo, Junyeob Kim, Hyuhng Joon Kim, Hyunsoo Cho, Hwiyeol Jo, Sang-Woo Lee, Sang goo Lee, and Taeuk Kim. 2022. Ground-truth labels matter: A deeper look into input-label demonstrations.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. LIMA: Less is more for alignment. In *Proceedings of NeurIPS*.

A Data Templates

1. Data Template for PACIT. Our proposed PACIT method takes the task definition, examples and instance input as the prompt. The model first generates the response to the auxiliary classification task and corresponding action of the provided examples. Based on the quiz result and action to be taken, the model then produces the outputs for the instance input for the given task.

Task Definition: {{definition}}
Example 1
- **Input: {{exp.input}}**
- **Output: {{exp.output}}**

Example 2
- **Input: {{exp.input}}**
- **Output: {{exp.output}}**

Evaluation Instance
- **Input: {{exp.input}}**

Classification
- **Classification result: {{Example 1 is correct/wrong and example 2 is correct/wrong.}}**
- **Generated action: {{I should learn from correct examples and avoid the mistakes in these wrong examples.}}**

Answering
- **Output: {{exp.output}}**

Figure 4: The data template used for PACIT method.

2. Data Template for Generating Examples with Self-Instruct. When generating positive and negative examples with the Self-instruct method, we randomly select four pairs of positive and negative examples in total from all examples of different tasks in the SuperNI dataset as in-context learning examples. We use ChatGPT (gpt-3.5-0613) to generate a positive and negative example pair based on the prompt shown in Figure 5.

Few-Shot Demonstrations:
Demonstrated Task Definition: {{definition}}
Positive Example
- **Input: {{exp.input}}**
- **Output: {{exp.output}}**

Negative Example
- **Input: {{exp.input}}**
- **Output: {{exp.output}}**

......
Generated Examples:
Task Definition: {{definition}}

Positive Example
- **Input: {{gen.input}}**
- **Output: {{gen.output}}**

Negative Example
- **Input: {{gen.input}}**
- **Output: {{gen.output}}**

Figure 5: The data template for generating positive and negative examples with the Self-instruct method.