Small Models are Valuable Plug-ins for Large Language Models

Canwen Xu[1*], **Yichong Xu**[2], **Shuohang Wang**[3], **Yang Liu**[3],
Chenguang Zhu, Julian McAuley[1]
[1]University of California, San Diego, [2]Character.AI, [3]Microsoft
[1]{cxu,jmcauley}@ucsd.edu, [2]yichong@character.ai,
[3]{shuowa,yaliu10}@microsoft.com

Abstract

Large language models (LLMs) such as GPT-3 and GPT-4 are powerful but their weights are often publicly unavailable and their immense sizes make the models difficult to be tuned with common hardware. As a result, effectively tuning these models with large-scale supervised data can be challenging. As an alternative, In-Context Learning (ICL) can only use a small number of supervised examples due to context length limits. In this paper, we propose Super In-Context Learning (SuperICL) which allows black-box LLMs to work with locally fine-tuned smaller models, resulting in superior performance on supervised tasks. Our experiments demonstrate that SuperICL can improve performance beyond state-of-the-art fine-tuned models while addressing the instability problem of in-context learning.[1]

1 Introduction

Large-scale pre-trained language models, such as GPT-3 (Brown et al., 2020) and GPT-4 (OpenAI, 2023), have demonstrated remarkable capabilities in a wide range of NLP tasks. Despite the impressive performance of these recently released models, their size can lead to difficulties in fine-tuning these models with supervised data, which is an effective way to adapt the models to specific tasks (Liu et al., 2019).

An alternative approach, In-Context Learning (ICL, Brown et al., 2020), involves concatenating a few labeled examples with the test input, enabling the model to learn from the context. However, ICL is limited by the maximum context length of the LLM, restricting the number of examples it can utilize. Consequently, while ICL can usually perform few-shot learning with 16 or 32 examples, it cannot fully exploit supervised data when there are hundreds or thousands of examples.

To address these limitations, we propose Super In-Context Learning (SuperICL), a novel approach that enables black-box language models (e.g., InstructGPT, GPT-4) to work with locally fine-tuned smaller models (e.g., RoBERTa, Liu et al., 2019), resulting in improved performance on supervised tasks. SuperICL is designed to overcome the challenges of poor performance and instability of ICL.

SuperICL builds on the strengths of ICL while mitigating its limitations. As shown in Figure 1, SuperICL leverages a combination of an LLM with smaller models, which act as plug-ins, to perform supervised tasks efficiently. Specifically, we use the plug-in model to predict labels with confidence for in-context examples and concatenate them with the input text and ground-truth labels as context. For test examples, we also add the plug-in model's prediction and confidence to the test input and let the LLM predict the final label and an explanation. As these plug-in models have been fine-tuned on the task-specific data, they serve as a bridge between the large pre-trained model and the task-specific data, allowing for effective knowledge transfer and improved performance.

We conduct extensive experiments to evaluate the effectiveness of SuperICL on GLUE (Wang et al., 2019), a standard benchmark for natural language understanding. Our results show that SuperICL: (1) achieves superior performance compared to state-of-the-art fine-tuned models and LLMs; (2) addresses the instability problem of ICL by allowing the plug-in models to absorb task-specific information while leaving the LLMs to focus on more general language understanding; (3) enhances the capabilities of plug-in models such as improving their cross-lingual transfer ability; (4) provides interpretability via the LLM by providing explanations for why it overrides predictions made by plug-in models. Our findings demonstrate the potential of combining large and small, cloud and local models, shedding light on a promising new

[*]Work done at Microsoft.
[1]Code available at https://github.com/JetRunner/SuperICL

Findings of the Association for Computational Linguistics ACL 2024, pages 283–294

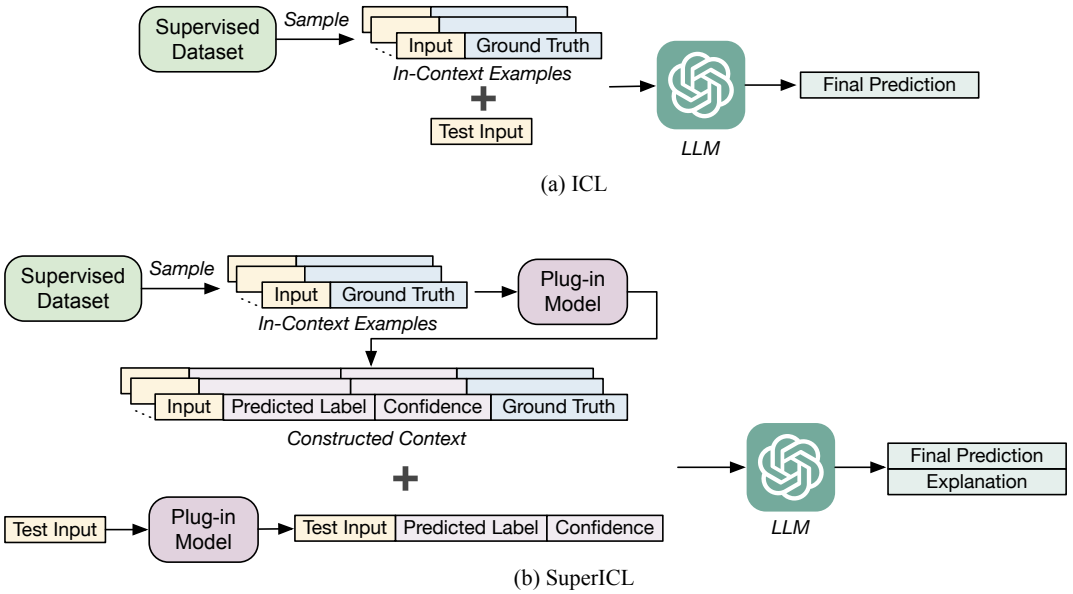

Figure 1: The workflow of ICL and SuperICL. There are three steps in SuperICL: (1) A context is constructed by randomly sampling from the training data and incorporating the plug-in model's predictions, including predicted labels and their corresponding confidence scores. (2) The test input is concatenated after the context, with the plug-in model's prediction attached. (3) Finally, a language model generates the final prediction along with an optional explanation.

paradigm for utilizing large language models. Our work also shows that an LLM's decision-making can benefit from explicitly including confidence of called plug-in models, which may be promising for complex task planning with LLMs (Wu et al., 2023a; Shen et al., 2023; Singh et al., 2023).

2 Related Work

In-Context Learning Originally proposed in the GPT-3 paper (Brown et al., 2020), In-Context Learning (ICL) is considered as a new paradigm that exploits LLMs on new tasks without updating the parameters of the model. It prepends few-shot training examples before the test input as a prompt, to enable large language models to find patterns and "learn" to predict. There have been successful applications of ICL in downstream tasks, such as machine translation (Lin et al., 2021; Agrawal et al., 2022) and data generation (Ye et al., 2022).

Despite its success in few-shot learning, a major drawback of ICL is instability. The performance of ICL is sensitive to the selected in-context examples (Zhao et al., 2021) and even their order (Lu et al., 2022). Based on these discoveries, there is a line of studies focused on constructing the context. LM-BFF (Gao et al., 2021) and KATE (Liu et al., 2022) select training examples that are semantically similar to the test example. Another line of works (Su et al., 2022; Levy et al., 2022; Ye et al., 2023) focus on mining diverse and representative examples from a training set. Zhang et al. (2022) utilizes active learning and reinforcement learning to select examples for ICL. Self-adaptive ICL (Wu et al., 2022, 2023b) proposes a two-stage search framework to obtain the optimal in-context examples for each test input without using a separate validation set. Different from these works, SuperICL demonstrates that smaller models can be integrated into large language models for supervised tasks. Although it is orthogonal to these prior works, by fine-tuning the plug-in model with the entire training set, SuperICL reduces the necessity for selecting the optimal examples from the training set.

Moreover, prior studies also investigate how to prepare language models for ICL. Zhao et al. (2021) propose calibration with an empty test input to reduce the influence of the label distribution and ordering. MetaICL (Min et al., 2022a) meta-trains the language model to generalize to unseen tasks for better ICL performance. Chen et al. (2022) propose four self-supervised objectives as intermediate tasks, to improve performance of language models on ICL. Notably, both Min et al. (2022a) and Chen et al. (2022) require updating the weights, thus are not applicable to larger black-box models

Algorithm 1 Super In-Context Learning (SuperICL)

Require: Training set $D = \{(x_1, y_1), ..., (x_n, y_n)\}$, LLM M, a small pre-trained language model P
Ensure: Predicted label y_t and optional explanation e_t

```
Fine-tune P on D to obtain the fine-tuned plug-in model P'
Randomly sample a set of examples (x_i, y_i) from D to be the set of in-context examples D'
for each example (x_i, y_i) in D' do
    Predict y'_i and c_i with P' where y'_i is the predicted label and c_i is the confidence score
end for
Construct the context C by concatenating all (x_i, y'_i, c_i, y_i)
for each test example x_t do
    Predict y'_t and c_t with P' for the test example
    Formulate complete input I = C ⊕ (x_t, y'_t, c_t) where ⊕ denotes concatenation
    Use M to predict y_t from I
    (Optional) If y_t ≠ y'_t, ask M to generate an explanation e_t for overriding the prediction of P'
end for
```

like GPT-3/4.

Besides studies aiming to improve ICL's performance, some studies have analyzed the underlying mechanism of ICL. Min et al. (2022b) find that the label space, the distribution of the input text, and the overall format of the sequence are the key factors to ICL's performance. They also claim that the ground labels are not significant to the performance of ICL, but this conclusion is contradicted by a later study (Yoo et al., 2022). Additionally, prior studies suggest ICL could be implicitly performing Bayesian inference (Xie et al., 2022) or gradient descent (Akyürek et al., 2022; von Oswald et al., 2022; Dai et al., 2022).

Language Model Plug-ins Large language models can exploit external tools to improve their capabilities. Toolformer (Schick et al., 2023) introduces special symbols that allow the large language models to call external APIs to complete tasks. Visual ChatGPT (Wu et al., 2023a) plugs vision models into ChatGPT, allowing for multimodal generation. HuggingGPT (Shen et al., 2023) uses ChatGPT to conduct task planning and select models according to their function descriptions available in Hugging Face, execute each subtask with the selected AI model, and summarize the response according to the execution results. Different from these works, our work is under a classic supervised learning and demonstrates that even tasks like text classification, which is sometimes considered "solved" by smaller language models, can still benefit from combination with a large language model.

3 Super In-Context Learning

Super In-Context Learning (SuperICL) combines LLMs with locally fine-tuned smaller models, allowing them to work together to improve performance on supervised tasks. The smaller models act as plug-ins, providing task-specific knowledge and predictions, while the large pre-trained models focus on general language understanding. The overall workflow of SuperICL is shown in Figure 1 and the complete algorithm is depicted in Algorithm 1.

Plug-in Model Fine-tuning The first step in the SuperICL process is fine-tuning a small NLP model, e.g., RoBERTa (Liu et al., 2019), on task-specific labeled data. This fine-tuning process on the entire training data is made possible due to the smaller size of the model and its local accessibility. This is in contrast to ICL, whose usage of labeled data is severely limited by the LLM's context length. The fine-tuned small model is then integrated as a plug-in for the LLM in subsequent steps as follows.

Context Construction Next, a context is constructed for the LLM to utilize the task-specific knowledge provided by the smaller model. This context consists of a set of examples randomly sampled from the training data, along with their corresponding predictions by the smaller plug-in model. The predictions include both the predicted labels and their associated confidence scores. An example is shown in Table 1.

On one hand, by incorporating the predicted labels by the plug-in model, the LLM can better understand the relationship among the input exam-

(a) Context	`Sentence 1:` Federal agent Bill Polychronopoulos said it was not known if the man, 30, would be charged. `Sentence 2:` Federal Agent Bill Polychronopoulos said last night the man involved in the Melbourne incident had been unarmed. `RoBERTa-Large Prediction:` equivalent (Confidence: 0.51) `Label:` not_equivalent `Sentence 1:` Five more human cases of West Nile virus, were reported by the Mesa County Health Department on Wednesday. `Sentence 2:` As of this week, 103 human West Nile cases in 45 counties had been reported to the health department. `RoBERTa-Large Prediction:` not_equivalent (Confidence: 0.98) `Label:` not_equivalent ...
(b) Test Input	`Sentence 1:` Cooley said he expects Muhammad will similarly be called as a witness at a pretrial hearing for Malvo. `Sentence 2:` Lee Boyd Malvo will be called as a witness Wednesday in a pretrial hearing for fellow sniper suspect John Allen Muhammad. `RoBERTa-Large Prediction:` equivalent (Confidence: 0.82)
(c) Label Prediction	`Label:` not_equivalent
(d) Explanation	`Explanation for overriding the prediction:` The two sentences are talking about different people, John Allen Muhammad and Lee Boyd Malvo, and thus the prediction should be not_equivalent.

Table 1: An example of the constructed context and inference procedure from the MRPC dataset. We first construct the context by sampling from the supervised dataset and attach the plug-in model's predictions. Then, for each test example, we ask the large language model to predict the label based on the input and the plug-in model's prediction. We use a prompt to ask the model to explain the decision if the label predicted by the plug-in model is overridden. The text field names (e.g., Sentence 1) are the original field names provided in the dataset.

ples, ground-truth labels and the plug-in model's expertise. This will help the LLM in the subsequent decision-making process to produce final predictions. On the other hand, confidence scores provide a measure of the plug-in model's uncertainty in its predictions. By incorporating these scores in the context, the LLM can trust predictions where the plug-in model is highly confident and be more cautious when the plug-in model is uncertain. Furthermore, confidence scores can help guide the LLM's attention towards in-context examples that are more challenging, enabling it to learn from these difficult cases and potentially improve its overall performance on the task.

In summary, by considering the predicted label and the associated confidence of the plug-in model, the LLM decides whether to follow the given predictions or to rely on its own understanding of the task, leading to more accurate predictions overall.

Inference Once the context has been constructed, the test input (an example shown in Table 1(b)) is concatenated with the context, forming a complete input for the large language model. The plug-in model's prediction for the test input, including the predicted label and confidence score, is also attached to the input. Thus, the LLM's input includes the context, test input, and plug-in model's prediction. The LLM then generates a final prediction for the test input, as shown in Table 1(c). Optionally, as shown in Table 1(d), the LLM can also provide an explanation for its prediction, giving insight into why it chose to override or follow the plug-in model's prediction. This additional interpretability can be valuable for understanding the decision-making process of the combined SuperICL model.

4 Experiments

4.1 Experimental Settings

Benchmarks We focus on the full supervised setting, where we have access to the entire training set. We conduct experiments on two widely-used benchmarks: the GLUE benchmark (Wang et al., 2019) for natural language understanding tasks and the XNLI benchmark (Conneau et al., 2018) for zero-shot cross-lingual natural language inference tasks, where the models are trained on English and tested on other languages. Our goal is to examine the learning ability of SuperICL on standard benchmarks and whether it can empower smaller models with its cross-lingual transfer capability. For both ICL and SuperICL, we only

Method	MNLI-m	MNLI-mm	SST-2	QNLI	MRPC	QQP	CoLA	RTE	Avg.
InstructGPT ICL	80.80	82.39	91.39	80.52	60.05	81.64	60.51	86.28	81.32
RoBERTa-Large	88.68	89.47	96.44	94.07	83.09	92.11	64.55	87.00	88.68
SuperICL	**89.31**	**89.61**	**96.79**	**94.16**	**86.03**	**92.14**	**64.57**	**87.73**	**89.90**

Table 2: Experimental results on GLUE (Wang et al., 2019) development set. The metric for CoLA is Matthews Correlation and all other tasks use accuracy. The improvement of SuperICL over RoBERTa-Large is statistically significant (3 runs, $p < 0.05$).

Method	MNLI	SST-2	MRPC
Ensemble ICL+RoBERTa	73.22	95.87	76.96
Self-adaptive ICL	85.50	91.97	73.52
SuperICL	**89.31**	**96.79**	**86.03**

Table 3: Comparison with ensemble InstructGPT ICL and RoBERTa-Large and a state-of-the-art ICL approach, Self-adaptive ICL (Wu et al., 2022). We take the average of the probabilities for the ensemble baseline. We use the normalized probability of each candidate label token for InstructGPT ICL. The results for Self-adaptive ICL are run by us with InstructGPT, thus are higher than reported in Wu et al. (2022).

consider the prediction to be correct when the generated label matches the predefined label exactly. For analytical experiments, we evaluate the model on a subset of GLUE consisting of three representative tasks: MNLI, SST-2 and MRPC, due to budget constraints.

Large Language Model and Plug-ins We use OpenAI's `text-davinci-003` language model, also known as InstructGPT. For the GLUE benchmark, we use RoBERTa-large (Liu et al., 2019) as the plug-in model. For the XNLI benchmark, we use XLM-V (Liang et al., 2023), as the plug-in model. Both models are fine-tuned on their respective tasks to serve as plug-ins for SuperICL. For GLUE tasks, we randomly select 32 examples from the training set. For XNLI, as the input is multilingual, the BPE tokenizer used in InstructGPT results in a longer token sequence. Thus, we use at most 16 examples for each language. Note that for some languages (e.g., Thai), the in-context examples are fewer than 16, as we fit as many examples as possible to the maximum allowed sequence length of 4,096 of InstructGPT. For our main experiments and all analysis experiments, we compare the performance of SuperICL with ICL (Brown et al., 2020) on the same selection of in-context examples and the predictions made by the plug-in models alone.

Lang.	InstructGPT ICL	XLM-V	SuperICL
en	74.03	83.55	**83.87**
ar	60.15	70.78	**72.28**
bg	67.64	77.09	**77.74**
de	71.78	75.23	**80.28**
el	65.85	72.73	**74.29**
es	76.79	77.07	**81.38**
fr	74.99	77.01	**77.47**
hi	56.29	69.62	**70.02**
ru	65.39	73.53	**76.85**
sw	56.13	67.43	**68.94**
th	57.03	68.90	**69.36**
tr	66.01	72.34	**72.63**
ur	51.18	**63.57**	57.90
vi	62.91	72.91	**74.45**
zh	67.90	73.75	**74.21**
Avg.	64.94	73.03	**74.11**

Table 4: Experimental results on the cross-lingual XNLI (Conneau et al., 2018) test set. The metric is accuracy.

4.2 Main Results

GLUE As shown in Table 2, SuperICL outperforms both InstructGPT ICL and the plug-in model RoBERTa-Large with an average advantage of 8.58 and 1.22 on GLUE, respectively. It is worth noting that SuperICL consistently outperforms the baselines on all tasks, which makes it a reliable choice that would not compromise the performance of the plug-in model. As shown in Table 3, our approach also outperforms baselines including ensemble ICL and RoBERTa, and a state-of-the-art ICL method (Wu et al., 2022). Additional results with DeBERTa (He et al., 2021) and LLaMA-2 (Touvron et al., 2023) (shown in Appendix A) further verify the effectiveness of SuperICL with alternative small and large models.

XNLI For XNLI, as presented in Table 4, while XLM-V (Liang et al., 2023) is specifically designed for multilingual tasks, combining it with InstructGPT can still lead to significant improvements in most languages. However, SuperICL fails to enhance the performance of XLM-V for Urdu. It is worth mentioning that InstructGPT ICL also ex-

	Components			MNLI	SST-2	MRPC
	(a) Ctxt.	*(b) Conf.*	*(c) Ref.*			
InstructGPT ICL				80.80	91.39	60.05
RoBERTa-Large				88.68	96.44	83.09
(1)	✓	✓	✗	81.23	92.43	65.69
(2)	✓	✗	✓	88.75	96.67	83.09
(3)	✗	✓	✓	88.89	96.44	83.59
(4)	✗	✗	✓	88.84	96.44	83.58
	✓	✓	✓	**89.31**	**96.79**	**86.03**

Table 5: Experimental results of the ablation study. *(a) Ctxt.* means the in-context examples from the training set; *(b) Conf.* represents the plug-in model's confidence score; *(c) Ref.* means whether we use the plug-in model's prediction for the test input.

Method	MNLI	SST-2	MRPC
%Overriden	0.22%	0.23%	12.50%
Overridden Accuracy	81.81%	100.00%	64.71%

Table 6: Statistics of overridden predictions. "%Overridden" indicates the percentage of final predictions that differ from the plug-in model's predictions, out of the total number of examples. "Overridden Accuracy" represents the percentage of correct predictions among the overridden ones.

hibits poor performance for Urdu, implying that InstructGPT may lack ability for low-resource languages like Urdu. This is also consistent with recent analysis on the multilinguality of InstructGPT/ChatGPT (Lai et al., 2023). Additionally, since the BPE tokenizer used in InstructGPT yields more tokens for non-Latin languages, the number of in-context examples is limited, adversely affecting the model's performance. We believe that subsequent GPT models that employ a multilingual tokenizer, train on more non-English data, and have a longer maximum context can achieve even better performance for cross-lingual SuperICL.

4.3 Ablation Study

We conduct an ablation study to understand the effect of each component in SuperICL. We investigate the performance of the three components in SuperICL: (a) Context, which comprises in-context examples; (b) The confidence scores of the plug-in model in both in-context examples and test input; (c) The plug-in model's prediction for the test input.

The experimental results are shown in Table 5: (1) We first attempt to remove the plug-in model's prediction for the test input. This has a significant negative impact on the performance of ICL as it

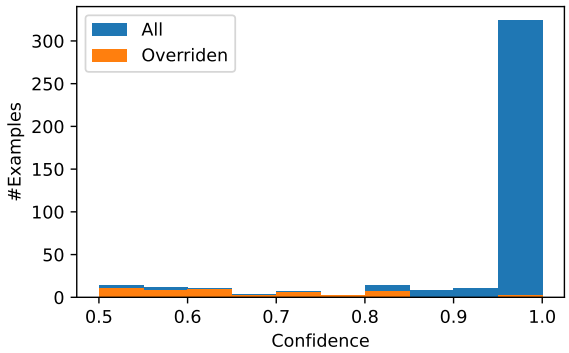

Figure 2: Effect of plug-in model confidence for overrides. The figure is the distribution of RoBERTa confidence for all examples (blue) and examples with a final prediction overridden by InstructGPT (orange) on MRPC.

creates a mismatch between in-context examples and test input. Interestingly, even though we remove the plug-in model for test input, SuperICL can still outperform ICL. We hypothesize this is due to an in-context effect similar to knowledge distillation (Hinton et al., 2015), which transfers task knowledge from the fine-tuned RoBERTa to InstructGPT. (2) We attempt to remove the confidence scores from SuperICL which results in a decrease in its performance. This is because InstructGPT becomes unaware of the uncertainty of RoBERTa, and as a result, it is unable to determine when to override the prediction. Also, similar to removing the softmax score from knowledge distillation, removing the confidence score makes knowledge transfer less effective. (3) When removing all in-context examples, SuperICL is essentially doing zero-shot inference for the test input. Although there is a slight improvement over RoBERTa, we can see adding in-context examples help SuperICL learn to calibrate the confidence and override RoBERTa's prediction. Also similar to ICL versus zero-shot inference, adding in-context examples helps InstructGPT to improve its own task-specific performance. (4) Further removing confidence scores from zero-shot inference also slightly decreases the performance.

4.4 Analysis on Prediction Overrides

We also analyze the statistics of predictions overridden by InstructGPT, as displayed in Table 6. A significant difference can be observed between various datasets. On both MNLI and SST-2, InstructGPT overrides only a minimal portion of examples (approximately 0.2%), but with high accu-

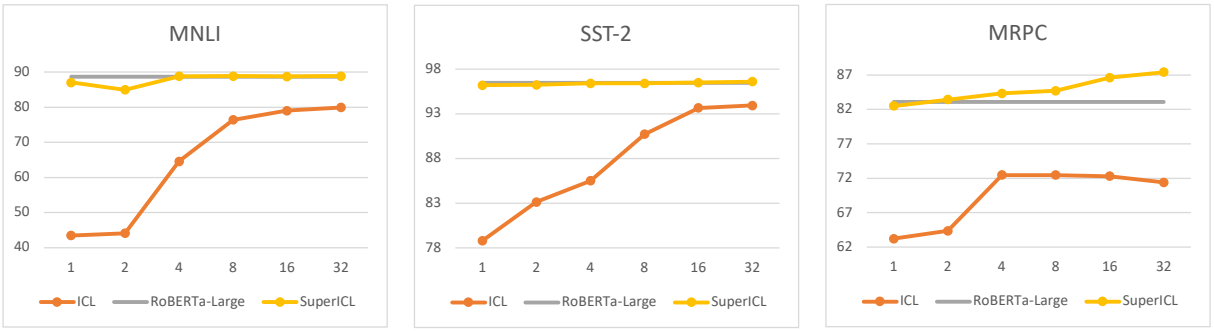

Figure 3: Effect of number of examples on the performance of ICL and SuperICL. The results are averages of three runs.

racy. Conversely, InstructGPT overrides a substantial percentage of 12.5% of predictions made by RoBERTa, although with lower accuracy. These findings suggest that the override behavior of SuperICL is heavily reliant on the specific dataset and the performance of the plug-in model.

To gain insights into the decision-making process of the LLM in overriding the predictions of the plug-in model, we examine the distribution of confidence levels exhibited by RoBERTa and the extent to which InstructGPT overrides them, as shown in Figure 2. The findings reveal a pattern where InstructGPT tends to override predictions when the plug-in model's confidence is low. This behavior supports our motivation and indicates that InstructGPT recognizes the uncertainty associated with the plug-in model's predictions via the confidence scores.

We further verify if SuperICL is naively flipping low-confidence predictions of RoBERTa. On low-confidence examples (defined as confidence $<$ 0.7), SuperICL's accuracy is 75.61% while a strategy of always flipping (flipping all low-confidence predictions) gives an accuracy of 63.40%. This proves that SuperICL is not naively flipping low-confidence predictions.

4.5 Analysis on Example Selection

We compare ICL and SuperICL by analyzing the sensitivity of different in-context examples. To ensure a fair comparison, we randomly sample five different batches of in-context examples for each dataset using different random seeds and ensure that the same in-context examples are used for both ICL and SuperICL.

Our results, as shown in Table 7 show that ICL demonstrates a larger variance compared to SuperICL, especially on MRPC, and its performance is drastically affected by the selection of in-context examples. On the other hand, SuperICL consistently outperforms both ICL and RoBERTa, while maintaining a more stable performance.

	Method	Experiment #					Var.
		1	2	3	4	5	
MNLI	ICL	80.80	81.26	79.74	81.26	80.79	0.39
	RoBERTa	88.68	88.68	88.68	88.68	88.68	-
	SuperICL	**89.31**	**88.94**	**88.79**	**89.17**	**88.78**	0.06
SST-2	ICL	91.39	94.04	94.38	93.12	93.46	1.35
	RoBERTa	96.44	96.44	96.44	96.44	96.44	-
	SuperICL	**96.79**	**96.56**	**96.56**	**96.56**	**96.56**	0.01
MRPC	ICL	60.05	73.53	73.28	73.28	65.44	37.50
	RoBERTa	83.09	83.09	83.09	83.09	83.09	-
	SuperICL	**86.03**	**87.99**	**87.75**	**84.31**	**86.52**	2.20

Table 7: Accuracy and variance of ICL and SuperICL with example selections randomly sampled with different seeds.

We argue that SuperICL's improved stability is due to the added reference prediction from the plug-in model, which has been trained on a large supervised training set. This helps the LLM to focus on learning to correct RoBERTa's predictions and override them when needed, minimizing the differences caused by different in-context examples.

4.6 Analysis on Number of Examples

We compare the effect of the number of examples on the performance of ICL and SuperICL in Figure 3. As shown, SuperICL can already achieve the best accuracy with merely 2 or 4 examples, while ICL requires more examples to achieve reasonable accuracy. Also, on the more difficult dataset MRPC, SuperICL can improve the performance of RoBERTa by a large margin with more in-context examples present.

(1) MNLI `Premise`: Then he sobered. `Hypothesis`: He was always sober. `RoBERTa Prediction`: neutral (Confidence: 0.7) ✗ `Final Prediction`: contradiction ✓ `Explanation`: The premise states that he sobered, implying that he was not sober before, while the hypothesis states that he was always sober, which contradicts the premise.
(2) MNLI `Premise`: Without the discount, nobody would buy the stock. `Hypothesis`: Nobody would buy the stock if there was a discount. `RoBERTa Prediction`: contradiction (Confidence: 0.6) ✓ `Final Prediction`: entailment ✗ `Explanation`: The premise and hypothesis are essentially the same statement, but with the premise being a negative statement and the hypothesis being a positive statement. The model interprets this as a contradiction, but it should be an entailment.
(3) SST-2 `Sentence`: no screen fantasy-adventure in recent memory has the showmanship of clones' last 45 minutes. `RoBERTa Prediction`: negative (Confidence: 0.54) ✗ `Final Prediction`: positive ✓ `Explanation`: The sentence is actually a positive statement, praising the last 45 minutes of the movie for its showmanship. The prediction was flipped because the sentence also contains the word "no" which can be interpreted as a negative sentiment.

Table 8: Examples of SuperICL's predictions and explanations. The contexts are omitted. The prompts here are modified for clarity. The original prompt template is shown in Table 1. The correct and incorrect predictions are marked with ✓and ✗, respectively.

Label Distribution	MNLI	SST-2	MRPC
Training Set (provided)	33:33:33	44:56	67:33
Test Set	32:33:34	49:51	68:32
SuperICL	89.31	**96.79**	86.03
SuperICL + Label Stats	89.31	96.67	**87.25**

Table 9: Experimental results of label distribution-aware SuperICL. By including training set label distribution, the LLM makes predictions leaning towards the majority class. This can be favorable if the test data has the same data distribution as the training set.

4.7 Label Distribution-Aware SuperICL

Classification models also learn data distribution and label biases from training data (Zhang et al., 2017). We are curious if the behavior of SuperICL can be affected when we explicitly inform the model of the data distribution in a dataset. We conduct additional experiments by providing the label distribution of the *training* set to the LLM, by adding a prompt at the beginning of the original prompt, e.g., "Label distribution: 'equivalent': 67.45%; 'not_equivalent': 32.55%."

The results are shown in Table 9. For test sets that have the same label distribution, adding such statistics can indeed improve the performance of the model on MRPC, which is unbalanced and has the same label distribution for training and test set. On the contrary, for SST-2, the test set has a different label distribution from the training set and adding training set label distribution seems to have a slight negative effect on the performance.

5 Case Study

We conduct a case study to better understand the behavior of SuperICL, with three examples presented in Table 8. We find that even without any explicit task instruction, InstructGPT demonstrates the ability to comprehend the tasks and explain its own reasoning. In the first example from Table 8, InstructGPT effectively grasps the implication in the premise that "he" was not sober. However, in the second example, InstructGPT incorrectly flips the prediction, possibly due to confusion caused by negation. This phenomenon has been recognized as a common flaw in LLMs, as noted by Hosseini et al. (2021) and Jang et al. (2023). In the last example, InstructGPT not only corrects RoBERTa's prediction successfully but also provides an analysis explaining why RoBERTa makes the wrong prediction.

6 Conclusion and Future Work

In this paper, we propose SuperICL, a simple yet effective method for combining a large language model API with a locally fine-tuned plug-in model. For future work, a theoretical analysis may be important to further reveal the internal mechanism of SuperICL.

Limitations

Additional Delay and Cost Since SuperICL involves serialized small and large models, the total inference delay equals to the sum of the inference delay of the two models. Also, calling the API of large language model can be expensive compared to using a locally deployed small model.

Adversarial Vulnerability As discussed in Appendix B, the vulnerability of the plug-in model to adversarial attacks can be inherited by SuperICL. Thus, when the plug-in model is under adversarial attack, the entire system could underperform ICL.

Limited Evaluation Tasks Due to space and budget limit, we only investigate text classification in this paper. However, it would be interesting to also look into generation tasks such as text summarization, question answering, and semantic parsing. Also, some datasets that have a larger impact for evaluating LLMs, like GSM8K, are missing in this paper, mainly due to the low performance of small models.

Broader Impact

As a technique that combines large and small language models for improved predictions, SuperICL shares the potential social biases of language models. While our approach is not likely to amplify these biases compared to other methods, it is important to investigate whether SuperICL has any effect on increasing or decreasing them. Furthermore, incorporating small models as plug-ins to the inference of large language models may lead to a slightly higher carbon footprint, resulting in a negative environmental impact. Therefore, practitioners should carefully consider the trade-offs between performance gains and environmental costs when using SuperICL.

References

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

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2022. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*.

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 *NeurIPS*.

Mingda Chen, Jingfei Du, Ramakanth Pasunuru, Todor Mihaylov, Srini Iyer, Veselin Stoyanov, and Zornitsa Kozareva. 2022. Improving in-context few-shot learning via self-supervised training. In *NAACL-HLT*, pages 3558–3573. Association for Computational Linguistics.

Alexis Conneau, Guillaume Lample, Ruty Rinott, Adina Williams, Samuel R Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Zhifang Sui, and Furu Wei. 2022. Why can gpt learn in-context? language models secretly perform gradient descent as meta optimizers. *arXiv preprint arXiv:2212.10559*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *ACL-IJCNLP*, pages 3816–3830. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R. Devon Hjelm, Alessandro Sordoni, and Aaron C. Courville. 2021. Understanding by understanding not: Modeling negation in language models. In *NAACL-HLT*, pages 1301–1312. Association for Computational Linguistics.

Joel Jang, Seonghyeon Ye, and Minjoon Seo. 2023. Can large language models truly understand prompts? a case study with negated prompts. In *Transfer Learning for Natural Language Processing Workshop*, pages 52–62. PMLR.

Viet Dac Lai, Nghia Trung Ngo, Amir Pouran Ben Veyseh, Hieu Man, Franck Dernoncourt, Trung Bui, and Thien Huu Nguyen. 2023. Chatgpt beyond english: Towards a comprehensive evaluation of large language models in multilingual learning. *arXiv preprint arXiv:2304.05613*.

Itay Levy, Ben Bogin, and Jonathan Berant. 2022. Diverse demonstrations improve in-context compositional generalization. *arXiv preprint arXiv:2212.06800*.

Davis Liang, Hila Gonen, Yuning Mao, Rui Hou, Naman Goyal, Marjan Ghazvininejad, Luke Zettlemoyer, and Madian Khabsa. 2023. Xlm-v: Overcoming the vocabulary bottleneck in multilingual masked language models. *arXiv preprint arXiv:2301.10472*.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, et al. 2021. Few-shot learning with multilingual language models. *arXiv preprint arXiv:2112.10668*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for gpt-3? In *DeeLIO@ACL*, pages 100–114. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

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 *ACL*, pages 8086–8098. Association for Computational Linguistics.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022a. Metaicl: Learning to learn in context. In *NAACL-HLT*, pages 2791–2809. Association for Computational Linguistics.

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? In *EMNLP*, pages 11048–11064. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*.

OpenAI. 2023. Gpt-4 technical report.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface. *arXiv preprint arXiv:2303.17580*.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. 2023. Progprompt: Generating situated robot task plans using large language models. In *ICRA*, pages 11523–11530. IEEE.

Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, et al. 2022. Selective annotation makes language models better few-shot learners. *arXiv preprint arXiv:2209.01975*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2022. Transformers learn in-context by gradient descent. *arXiv preprint arXiv:2212.07677*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*. OpenReview.net.

Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. 2023a. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*.

Zhenyu Wu, YaoXiang Wang, Jiacheng Ye, Jiangtao Feng, Jingjing Xu, Yu Qiao, and Zhiyong Wu. 2023b. Openicl: An open-source framework for in-context learning. *arXiv preprint arXiv:2303.02913*.

Zhiyong Wu, Yaoxiang Wang, Jiacheng Ye, and Lingpeng Kong. 2022. Self-adaptive in-context learning. *arXiv preprint arXiv:2212.10375*.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. In *ICLR*. OpenReview.net.

Jiacheng Ye, Jiahui Gao, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2022. Progen: Progressive zero-shot dataset generation via in-context feedback. In *EMNLP (Findings)*, pages 3671–3683. Association for Computational Linguistics.

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

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. In *EMNLP*, pages 2422–2437. Association for Computational Linguistics.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *ICLR*. OpenReview.net.

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022. Active example selection for in-context learning. In *EMNLP*, pages 9134–9148. Association for Computational Linguistics.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

A Experiments with Alternative Plug-in and Large LMs

To demonstrate that our method is able to generalize to another plug-in model and large LM, we conduct experiments with DeBERTa V3-Large (He et al., 2021) and LLaMA 2 (Touvron et al., 2023).

Experiments with DeBERTa We also attempt to replace the plug-in model with a state-of-the-art model, DeBERTa V3-Large (He et al., 2021) fine-tuned on the datasets. Our results, presented in Table 10, demonstrate that SuperICL continues to enhance the performance of state-of-the-art models, although the improvement is smaller compared to RoBERTa. This reduction may be attributed to the smaller capability gap between the small and large models. However, we expect that using an even more advanced large language model in the future will resolve this issue.

Method	MNLI	SST-2	MRPC
ICL	80.80	91.39	60.05
RoBERTa-Large	88.68	96.44	83.09
SuperICL + RoBERTa	**89.31**	**96.79**	**86.03**
DeBERTa V3-Large	90.49	96.56	90.44
SuperICL + DeBERTa	**90.76**	**96.79**	**90.93**

Table 10: Experimental results of SuperICL with different plug-in models.

Experiments with LLaMA-2 7B We conduct experiments with LLaMA-2 7B and the results are shown in Table 11. To summarize, LLaMA-2 7B can improve the performance of MRPC and SST-2. For MNLI, the performance of LLaMA-2 7B is very low with ICL and it cannot improve over the plug-in model for SuperICL (although it does not harm the performance). This may be due to its small size (7B parameters) and limited capability compared to GPT-3.5 (175B parameters).

Method	MNLI	SST-2	MRPC
LLaMA-2 7B ICL	47.45	92.89	68.14
RoBERTa-Large	88.68	96.44	83.09
LLaMA-2 7B SuperICL	88.68	**96.56**	**87.01**

Table 11: Experimental results of SuperICL with LLaMA-2 7B (Touvron et al., 2023).

B Analysis on Adversarial Robustness

Additionally, we analyze the adversarial robustness of SuperICL by testing it on ANLI (Nie et al., 2019). ANLI is a dataset for evaluating the robustness and generalization of natural language inference (NLI) models. It consists of 16,000 premise-hypothesis pairs that are categorized into three classes: entailment, contradiction, and neutral. The dataset is constructed with three rounds (R1, R2, and R3) and thus has three splits, with R3 being the most challenging and diverse. ANLI is collected using a human-and-model-in-the-loop training method, where human annotators act as adversaries and attempt to fool the model into misclassifying while still being understandable to other humans. This benchmark is designed to be challenging for language models including RoBERTa as RoBERTa is attacked in R2 and R3 of data construction.

As shown in Table 12, InstructGPT ICL is rather robust while RoBERTa-Large is vulnerable to adversarial attack. However, this directly has a negative impact on SuperICL. Although SuperICL achieves better performance than RoBERTa-Large, it underperforms ICL. This finding suggests that SuperICL's performance relies on the performance of the incorporated plug-in model and adversarial attack to the plug-in model could lead to a drastic performance drop for SuperICL.

Test Set # Examples	R1 1000	R2 1000	R3 1200	All 3200
ICL	59.50	52.40	52.58	54.69
RoBERTa-Large	41.60	27.40	24.58	30.78
SuperICL	56.10	42.70	44.17	47.44

Table 12: Zero-shot results on ANLI (Nie et al., 2019). ICL and SuperICL use in-context examples sampled from MNLI. The RoBERTa-Large model is fine-tuned on MNLI. R1, R2 and R3 denote the first, second and third round of adversarial attacks, respectively.

By cross-examine results from Table 2, 10, 11 and 12, we can conclude that the performance of

SuperICL correlates with the performance of the plug-in model and can degrade when the small model suffers from adversarial attacks. On the other hand, if the LLM performs exceptionally poorly on a task, it is unlikely that it can improve the performance of the plug-in model (XNLI on Urdu and the results on MNLI with LLaMA-2 7B).