# Feature-Adaptive and Data-Scalable In-Context Learning

Jiahao Li[1], Quan Wang[2], Licheng Zhang[1], Guoqing Jin[3], Zhendong Mao[1*]

[1]University of Science and Technology of China, Hefei, China
[2]MOE Key Laboratory of Trustworthy Distributed Computing and Service,
Beijing University of Posts and Telecommunications, Beijing, China
[3]People's Daily Online Co., Beijing, China
{jiahao66, zlczlc}@mail.ustc.edu.cn, wangquan@bupt.edu.cn
jinguoqing@people.cn, zdmao@ustc.edu.cn



## Abstract

In-context learning (ICL), which promotes inference with several demonstrations, has become a widespread paradigm to stimulate LLM capabilities for downstream tasks. Due to context length constraints, it cannot be further improved in spite of more training data, and general features directly from LLMs in ICL are not adaptive to the specific downstream task. In this paper, we propose a feature-adaptive and data-scalable in-context learning framework (FADS-ICL), which can leverage task-adaptive features to promote inference on the downstream task, with the supervision of beyond-context samples. Specifically, it first extracts general features of beyond-context samples via the LLM with ICL input form one by one, and introduces a task-specific modulator to perform feature refinement and prediction after fitting a specific downstream task. We conduct extensive experiments on FADS-ICL under varying data settings (4∼128 shots) and LLM scale (0.8∼70B) settings. Experimental results show that FADS-ICL consistently outperforms previous state-of-the-art methods by a significant margin under all settings, verifying the effectiveness and superiority of FADS-ICL. For example, under the 1.5B and 32 shots setting, FADS-ICL can achieve **+14.3** average accuracy from feature adaptation over vanilla ICL on 10 datasets, with **+6.2** average accuracy over the previous state-of-the-art method, and the performance can further improve with increasing training data. Code and data are publicly available at https://github.com/jiahaozhenbang/FADS-ICL.

## 1 Introduction

In recent years, increasingly large-scale pre-trained language models (LLMs) have emerged, which leads to a substantial cost for fine-tuning different models for each downstream task. Alternatively, in-context learning (ICL) has shown impressive performance on NLP downstream tasks without any modifications to LLMs. Specifically, ICL prepends several input-label pairs (demonstrations) to a test sample via a task-specific template and conducts prediction conditioned on this prompt, namely context, for better inference (Brown et al., 2020).

Most previous works focus on how to design the best prompt to steer its best performance. For example, some works aim to search for the most suitable templates for specific tasks (Sorensen et al., 2022; Prasad et al., 2023), while others focus on selecting the best demonstrations for each test sample (Liu et al., 2022; Rubin et al., 2022; Wang et al., 2023). There are even a few works that explore the order of demonstrations (Lu et al., 2022a; Wu et al., 2023). However, due to the context length constraints of LLMs, these ICL-based methods can only accept a limited number of samples as demonstrations put into the context, even if more labeled samples are available, which severely hinders performance on downstream tasks. Thus, what we need is **data scalability**, that is, the model can accept scalable labeled data to enhance model performance.

Recently, a few works have explored exploiting beyond-context samples, similar to KNN-LM (Khandelwal et al., 2020). Instead of putting all labeled samples into the context at once, they divide the inference process into two parts: sequentially obtaining the feature representation of each sample, and conducting prediction based on the relevance of the feature representation. Specifically, they compute nearest neighbors for the test sample based on the distance in the feature space and adjust prediction results accordingly. For example, kNN-prompt (Shi et al., 2022) performs the final prediction by interpolating the original predicted distribution with the distribution of nearby samples, while kNN-prompting (Xu et al., 2023) adopts voting by them directly. However, they ignore feature refinement for specific tasks and instead directly use general features obtained from the LLM designed for language modeling, which we refer to as

*Corresponding author: Zhendong Mao.

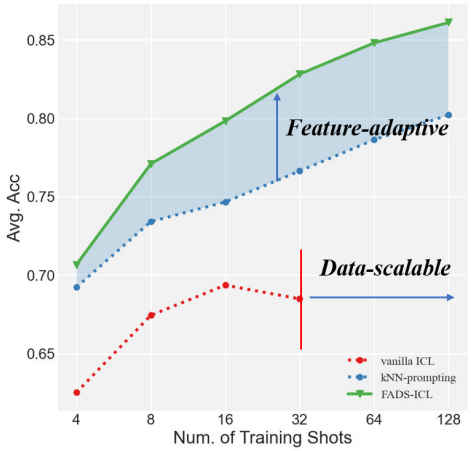


Figure 1: The effect of data scalability and feature adaptation on ICL. For data scalability, the performance of vanilla ICL cannot be further improved when exceeding a certain amount of data, but kNN-prompting and FADS-ICL can. For feature adaptation, FADS-ICL conducts feature refinement for specific tasks, so that it outperforms kNN-prompting using general features by large margins under all data settings.

the issue of **feature adaptation**. Such non-adaptive feature usage can also severely impair the performance on downstream tasks. From Figure 1, we can see that under the framework of ICL, data scalability and feature adaptation are both essential to performance on downstream tasks.

To this end, we proposed a feature-adaptive and data-scalable in-context learning framework (FADS-ICL), which can leverage task-adaptive features to promote inference on the downstream task, with the supervision of beyond-context samples. Specifically, FADS-ICL first extracts the general features of each sample by feeding them in the form of vanilla ICL to an LLM in turn. Then, these features and corresponding labels are used to supervise the training of a lightweight modulator for feature adaptation in the downstream task. Finally, during inference, this modulator can refine the general features to the task-adaptive features for test samples and perform the final prediction based on them. Here, FADS-ICL solves the issue of data scalability by feeding them in the form of ICL to the LLM one by one to extract general features, which does not need to put all samples into context. Besides, FADS-ICL also implements feature adaptation by further refining general features through a modulator for a specific downstream task.

To verify the effectiveness of FADS-ICL, we conduct evaluation experiments on 10 established datasets under varying amounts of labeled samples and LLM scales. The main experimental results show that FADS-ICL consistently outperforms state-of-the-art methods by a large margin under all data settings (4∼128 shots, refer to Figure 1) and all LLM scale settings (0.8B∼70B). Specifically, under the fixed 1.5B LLM and 32 shots, FADS-ICL can achieve **+14.3** average accuracy from feature adaptation over vanilla ICL on 10 datasets, with **+6.2** average accuracy over the previous state-of-the-art method, kNN-prompting. When growing to 128 shots, the improvement over vanilla ICL further increases to **+17.8**, showing its data scalability. In short, feature adaptation can improve downstream performance with a fixed amount of available data, while data scalability comes into play as more data becomes available.

Further, we systematically analyze the impact of three key settings in FADS-ICL, including modulators, selection of general features, and the number of demonstrations, which provides in-depth insights and the corresponding recommended configurations in FADS-ICL. The analysis experiment on different modulators reveals the importance of a parametric modulator that can promote feature adaptation in FADS-ICL. Besides, the analysis experiment on the selection of general features shows the hidden states containing rich semantics are more suitable for FADS-ICL than probability distribution. Further, our study on the role of demonstrations shows that demonstrations during feature extraction are vital to FADS-ICL while FADS-ICL is robust to the number of demonstrations.

Our contributions are summarized below: (1) We propose a feature-adaptive and data-scalable in-context learning framework, which can leverage task-adaptive features to promote inference on the downstream task, with the supervision of beyond-context samples. To our knowledge, this is the first time that feature adaptation has been considered in ICL. (2) Extensive experiments are conducted on 10 datasets under varying settings and show that FADS-ICL consistently outperforms previous state-of-the-art methods by a significant margin in almost all settings. (3) Detailed analytical experiments on key settings provide in-depth insights and recommended configurations in FADS-ICL.

## 2 Preliminary

### 2.1 ICL

ICL is actually a data-enhanced inference framework, which prepends demonstrations to a test sam-

ple and conducts prediction conditioned on this prompt for better inference. Specifically, assuming a train set $\{(x_i, y_i)\}$, it first needs to randomly sample several demonstrations $D = \{(x_i^D, y_i^D)\}$, of which the number $|D|$ is limited to the context length. Then, a task-specific template $\mathcal{T}(\cdot)$ (refer to Appendix F for details) is used to wrap up demonstrations along with a test sample $x_{test}$ followed by concatenation to obtain the prompt:

$$\mathcal{P} = \mathcal{T}(x_1^D, \mathcal{V}(y_1^D)) \oplus \mathcal{T}(x_2^D, \mathcal{V}(y_2^D)) \oplus \cdots \oplus \mathcal{T}(x_{|D|}^D, \mathcal{V}(y_{|D|}^D)) \oplus \mathcal{T}(x_{test}, *),$$

where $\mathcal{V}(\cdot)$ is a label verbalizer, mapping each label id to a token in the vocab, *e.g.* positive/negative or optimistic/pessimistic in sentiment classification. Finally, it takes the prompt as the input of the LLM, and computes the probability distribution by:

$$p(y|x) \propto p_\theta(\mathcal{V}(y)|\mathcal{P}), \tag{1}$$

with normalization, where $\theta$ denotes the parameters of the LLM. Absolutely, ICL can only make use of several demonstrations in context, in spite of the residual in the trainset, namely data non-scalability.

### 2.2 KNN-prompting

To implement data scalability, kNN-prompting tries to obtain probability distributions of residual samples in the same way and vote the test label by $k$ nearest neighbors based on distances between distributions, instead of obtaining the label directly from the distribution. Specifically, it first collects probability distributions of residual samples in the trainset by Equation 1. Then it computes distances between the distribution of the test sample and those of labeled samples based on KL divergence. Finally, $k$ nearest neighbors based on distances are responsible for voting on the test label. KNN-prompting implements data scalability by exploiting distributions of the residual in the trainset, but these distributions directly come from the freezing LLM designed for language modeling, that is, ignoring specific adaptation to downstream tasks.

## 3 FADS-ICL

For the sake of achieving both feature adaptation and data scalability, we propose FADS-ICL. In this section, we elaborate on the architecture of FADS-ICL, including a feature extractor and a lightweight task-specific modulator. The overall framework is depicted in Figure 2. Note that the train set in

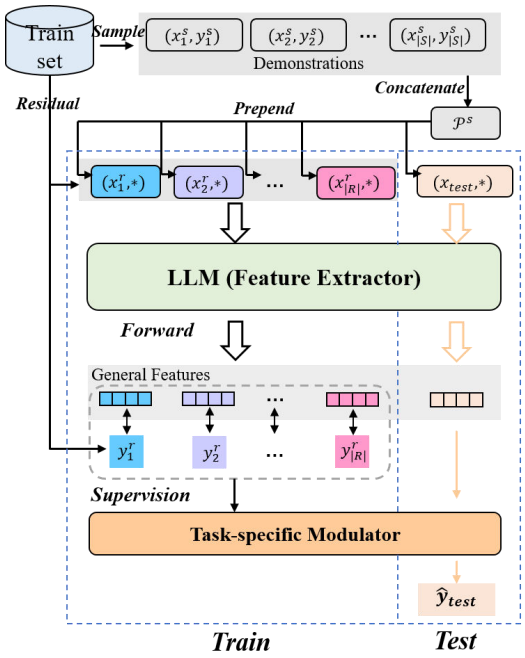


Figure 2: The overall framework of FADS-ICL.

FADS-ICL is split to a small demonstration set $S = \{(x_i^s, y_i^s)\}$ and a residual set $R = \{(x_i^r, y_i^r)\}$, with $D = S \cup R$ for fair comparison to ICL.

### 3.1 Feature Extractor

In FADS-ICL, the LLM is actually regarded as a feature extractor with the form of the vanilla ICL, which obtains general features for subsequent usage. Specifically, as in ICL, we first randomly choose a few labeled samples as demonstrations[1] $S$, making up the prompt $\mathcal{P}^s$ via a task-specific template. Then we feed each template-wrapped sample $x_i$ (labeled sample $x_i^r$ or test sample $x_{test}$) prepended with this identical prompt $\mathcal{P}^s$ into the LLM. The general feature of each sample can be obtained by a simple forward pass:

$$h(x_i) = f_\theta(\mathcal{T}(x_i, *)|\mathcal{P}^s), \tag{2}$$

where we choose the last hidden state from the last layer of the LLM as the common setting[2]. The obtained feature here serves language modeling rather than a specific downstream task, so-called the general feature, so it needs to undergo feature refinement before downstream usage.

[1]Note that FADS-ICL is not as sensitive to the number of demonstrations as ICL, of which one can refer to the subsection 5.3 for detailed analysis, and thus a few demonstrations are enough, *e.g.* one sample per class.

[2]More settings are discussed in subsection 5.2.

### 3.2 Task-specific modulator

A lightweight modulator is used to refine general features for feature adaptation on a specific downstream task and perform the final prediction. As downstream tasks are not visible to the LLM, inevitably there is the problem of task maladaptiveness. In other words, general features from the LLM are not well suited for direct downstream usage. Except for the heavy fine-tuning on the LLM, an efficient alternative solution is to utilize an extra lightweight module to perform feature adaptation to downstream tasks while keeping the LLM's parameters and generality. Specifically, We match the general features extracted from the residual samples in the training set with their corresponding labels, and they are taken as supervisory signals to train a lightweight modulator:

$$\phi = \arg\min_{\phi} \sum_{1 \le i \le |R|} \mathcal{L}(g_\phi(h(x_i^r)), y_i^r), \quad (3)$$

where the modulator[3] $g_\phi(\cdot)$ can be *Logistic Regression*, *Linear SVM*, *MLP*, *etc*, and the specific optimization objective $\mathcal{L}$ and algorithm depend on the specific choice of the modulator and possibly the specific task. After training, the modulator can be taken as a solver for the specific task. During inference, the modulator can refine general features for the specific task and map them to the label space without label verbalization in vanilla ICL:

$$p(y|x) = g_\phi(h(x)). \quad (4)$$

In short, FADS-ICL achieves data scalability by the split use of labeled data (*i.e.* using a fixed small part of the data as demonstrations and the residual scalable part for supervising the feature adaptation process) and multiple model calls. Furthermore, FADS-ICL achieves feature adaptation by refining general features into task-specific features via a well-trained task-specific modulator.

## 4 Experiments

### 4.1 Experimental Settings

**Datasets** Following kNN-prompting (Khandelwal et al., 2020), we established our experiments on 10 publicly available datasets, respectively four sentiment classification datasets: SST2 (Socher et al., 2013), MPQA (Wiebe et al., 2005), CR (Hu and Liu, 2004), and MR (Pang and Lee, 2005), two natural language inference datasets: CB (De Marneffe et al., 2019) and RTE (Dagan et al., 2005), three topic classification datasets: AGNews (Zhang et al., 2015), DBPedia (Zhang et al., 2015), and TREC (Voorhees and Tice, 2000), one subjectivity judgment dataset: SUBJ (Pang and Lee, 2004). The data statistics for all datasets are shown in Table 1.

**LLMs** To verify the effectiveness of FADS-ICL on different LLM scales, we consider a wide range of LLM scales, including GPT2 series (0.8B and 1.5B), Llama-1 series (7B, 13B, and 30B), and Llama-2 series (7B, 13B and 70B)[5]. GPT2-xl (1.5B) is used for most subsequent experiments unless explicitly stated.

**Other Settings or Details** For data setting, we consider varying training shots $m = \{4, 8, 16, 32, 64, 128\}$, which means the number of samples per class. We implement all modulators by scikit-learn (Buitinck et al., 2013), and conduct both the training and inference processes of modulators on the CPU. Note that even so, the total running time of the modulator is within a second [6] and it is negligible compared to the forward pass of the LLM. Besides, without special instructions, the lightweight modulator we choose is *Logistic Regression* with the last hidden states as general features, and the number of demonstrations used in FADS-ICL is one sample per class.

**Evaluation** Accuracy is the direct metric for all datasets. We run each experiment with five different seeds, and average accuracy and standard deviation are reported.

**Baselines** In addition to vanilla ICL, we also consider stronger baselines for fair comparison. kNN-prompt and kNN-prompting can both exploit beyond-context samples and improve the inference performance by interpolation or voting by k nearest neighbors. Besides, since sometimes ICL can not put all labeled samples as demonstrations into context, some methods have emerged to improve ICL by selecting the most appropriate demonstrations for each test sample during inference. We put the detailed comparison with demonstration selection methods in Appendix B.

[3] Modulator selection is discussed in subsection 5.1.

[4] https://github.com/BenfengXu/KNNPrompting

[5] Bitsandbytes for 8-bit quantification of Llama-2 (70B).

[6] Refer to subsection 4.2 for detailed overhead comparison.

| | SST2 | SUBJ | MPQA | AGNews | CB | CR | DBPedia | MR | RTE | TREC |
|---|---|---|---|---|---|---|---|---|---|---|
| **Avg. Len. of Samples** | 53.5 | 129.1 | 18.6 | 239.1 | 288.2 | 96.3 | 281.5 | 115.6 | 343.2 | 50.8 |
| **Num of Classes** | 2 | 2 | 2 | 4 | 3 | 2 | 14 | 2 | 2 | 6 |
| **Num. of Shots (TP)** | 20 (2%) | 12 (1%) | 39 (0%) | 3 (0%) | 2 (0%) | 14 (4%) | 1 (77%) | 14 (4%) | 4 (0%) | 8 (1%) |

Table 1: Data statistics for all used datasets. Num. of Shots (TP) denotes the Maximum number of training shots (per class) allowed by GPT2-xl, *i.e.* 1024 tokens of context. Inside the parentheses are Truncation Probability (TP).

| Setting&Methods | | SST2 | SUBJ | MPQA | AGNews | CB | CR | DBPedia | MR | RTE | TREC | AVG |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $m=4$ | **ICL** | 70.4$_{\pm 6.5}$ | 57.3$_{\pm 10.5}$ | 66.8$_{\pm 8.1}$ | 78.2$_{\pm 6.7}$ | 57.9$_{\pm 9.7}$ | 52.0$_{\pm 3.2}$ | 82.0$_{\pm 2.1}$ | 52.0$_{\pm 3.8}$ | 53.0$_{\pm 1.7}$ | 55.4$_{\pm 3.7}$ | 62.5$_{\pm 5.6}$ |
| | **kNN-prompt** | 60.1$_{\pm 9.7}$ | 76.4$_{\pm 7.8}$ | 55.7$_{\pm 8.7}$ | 84.0$_{\pm 5.3}$ | 42.1$_{\pm 1.0}$ | 83.3$_{\pm 6.0}$ | 85.2$_{\pm 2.8}$ | 78.2$_{\pm 7.7}$ | 53.5$_{\pm 2.6}$ | 54.7$_{\pm 3.6}$ | 67.3$_{\pm 5.5}$ |
| | **kNN-prompting** | 73.2$_{\pm 12.3}$ | 70.4$_{\pm 11.0}$ | 60.9$_{\pm 12.6}$ | 77.3$_{\pm 10.8}$ | 55.7$_{\pm 8.3}$ | 84.9$_{\pm 4.0}$ | 79.1$_{\pm 1.6}$ | 82.9$_{\pm 4.5}$ | 52.3$_{\pm 3.5}$ | 55.5$_{\pm 10.7}$ | 69.2$_{\pm 8.0}$ |
| | **FADS-ICL** | 68.7$_{\pm 6.6}$ | 83.3$_{\pm 4.3}$ | 61.6$_{\pm 8.3}$ | 78.0$_{\pm 6.1}$ | 52.9$_{\pm 16.7}$ | 83.2$_{\pm 9.9}$ | 91.1$_{\pm 2.8}$ | 76.1$_{\pm 10.3}$ | 53.4$_{\pm 0.3}$ | 58.4$_{\pm 11.9}$ | **70.7**$_{\pm 7.7}$ |
| $m=8$ | **ICL** | 73.3$_{\pm 11.6}$ | 64.1$_{\pm 11.3}$ | 72.7$_{\pm 9.0}$ | 78.2$_{\pm 6.7}$ | 57.9$_{\pm 9.7}$ | 66.2$_{\pm 16.7}$ | 82.0$_{\pm 2.1}$ | 72.2$_{\pm 13.9}$ | 53.0$_{\pm 1.7}$ | 54.8$_{\pm 4.2}$ | 67.4$_{\pm 8.7}$ |
| | **kNN-prompt** | 68.2$_{\pm 12.2}$ | 84.4$_{\pm 7.0}$ | 64.2$_{\pm 7.7}$ | 83.8$_{\pm 5.1}$ | 41.1$_{\pm 0.0}$ | 84.5$_{\pm 9.0}$ | 84.9$_{\pm 2.8}$ | 82.5$_{\pm 5.9}$ | 52.5$_{\pm 0.2}$ | 53.8$_{\pm 3.9}$ | 70.0$_{\pm 5.4}$ |
| | **kNN-prompting** | 81.4$_{\pm 10.8}$ | 76.0$_{\pm 5.4}$ | 71.4$_{\pm 7.9}$ | 84.3$_{\pm 1.2}$ | 45.7$_{\pm 7.7}$ | 83.5$_{\pm 2.9}$ | 89.9$_{\pm 1.4}$ | 84.5$_{\pm 4.0}$ | 53.6$_{\pm 6.1}$ | 63.7$_{\pm 9.4}$ | 73.4$_{\pm 5.7}$ |
| | **FADS-ICL** | 82.2$_{\pm 6.9}$ | 85.4$_{\pm 4.7}$ | 74.0$_{\pm 2.4}$ | 84.2$_{\pm 2.6}$ | 57.1$_{\pm 4.2}$ | 88.0$_{\pm 4.7}$ | 95.3$_{\pm 1.9}$ | 81.3$_{\pm 7.6}$ | 51.1$_{\pm 2.7}$ | 72.2$_{\pm 6.8}$ | **77.1**$_{\pm 4.4}$ |
| $m=16$ | **ICL** | 81.3$_{\pm 5.4}$ | 64.1$_{\pm 11.3}$ | 83.9$_{\pm 1.4}$ | 78.2$_{\pm 6.7}$ | 57.9$_{\pm 9.7}$ | 66.2$_{\pm 16.7}$ | 82.0$_{\pm 2.1}$ | 72.2$_{\pm 13.9}$ | 53.0$_{\pm 1.7}$ | 54.8$_{\pm 4.2}$ | 69.4$_{\pm 7.3}$ |
| | **kNN-prompt** | 79.2$_{\pm 13.6}$ | 85.9$_{\pm 5.0}$ | 75.2$_{\pm 6.7}$ | 83.4$_{\pm 6.3}$ | 42.1$_{\pm 1.0}$ | 82.3$_{\pm 7.3}$ | 85.8$_{\pm 2.1}$ | 79.7$_{\pm 7.6}$ | 52.4$_{\pm 0.2}$ | 54.7$_{\pm 4.1}$ | 72.1$_{\pm 5.4}$ |
| | **kNN-prompting** | 78.7$_{\pm 15.8}$ | 80.0$_{\pm 2.9}$ | 73.9$_{\pm 7.5}$ | 85.4$_{\pm 1.9}$ | 48.9$_{\pm 5.7}$ | 82.7$_{\pm 4.4}$ | 93.5$_{\pm 0.4}$ | 83.6$_{\pm 3.7}$ | 52.3$_{\pm 3.2}$ | 67.7$_{\pm 4.7}$ | 74.7$_{\pm 5.0}$ |
| | **FADS-ICL** | 86.6$_{\pm 7.8}$ | 89.1$_{\pm 2.3}$ | 79.2$_{\pm 1.7}$ | 86.5$_{\pm 2.1}$ | 63.9$_{\pm 3.4}$ | 88.7$_{\pm 3.4}$ | 97.2$_{\pm 0.6}$ | 82.5$_{\pm 3.4}$ | 48.8$_{\pm 3.3}$ | 75.7$_{\pm 6.0}$ | **79.8**$_{\pm 3.4}$ |
| $m=32$ | **ICL** | 81.3$_{\pm 5.4}$ | 64.1$_{\pm 11.3}$ | 75.2$_{\pm 8.8}$ | 78.2$_{\pm 6.7}$ | 57.9$_{\pm 9.7}$ | 66.2$_{\pm 16.7}$ | 82.0$_{\pm 2.1}$ | 72.2$_{\pm 13.9}$ | 53.0$_{\pm 1.7}$ | 54.8$_{\pm 4.2}$ | 68.5$_{\pm 8.1}$ |
| | **kNN-prompt** | 80.9$_{\pm 12.9}$ | 88.0$_{\pm 3.4}$ | 72.2$_{\pm 6.0}$ | 83.6$_{\pm 5.0}$ | 41.8$_{\pm 1.6}$ | 85.2$_{\pm 4.0}$ | 85.7$_{\pm 2.6}$ | 84.5$_{\pm 5.2}$ | 52.4$_{\pm 0.2}$ | 55.0$_{\pm 3.9}$ | 72.9$_{\pm 4.5}$ |
| | **kNN-prompting** | 83.9$_{\pm 7.6}$ | 83.0$_{\pm 1.6}$ | 71.8$_{\pm 7.9}$ | 85.3$_{\pm 1.5}$ | 55.0$_{\pm 6.4}$ | 86.4$_{\pm 3.3}$ | 94.5$_{\pm 0.5}$ | 84.5$_{\pm 1.2}$ | 51.5$_{\pm 5.8}$ | 70.6$_{\pm 2.9}$ | 76.6$_{\pm 3.9}$ |
| | **FADS-ICL** | 88.0$_{\pm 4.6}$ | 90.1$_{\pm 2.2}$ | 81.4$_{\pm 3.0}$ | 84.9$_{\pm 1.4}$ | 72.1$_{\pm 2.0}$ | 92.3$_{\pm 0.7}$ | 97.9$_{\pm 0.9}$ | 85.9$_{\pm 2.8}$ | 55.5$_{\pm 2.9}$ | 79.9$_{\pm 4.7}$ | **82.8**$_{\pm 2.5}$ |
| $m=64$ | **ICL** | 81.3$_{\pm 5.4}$ | 64.1$_{\pm 11.3}$ | 75.2$_{\pm 8.8}$ | 78.2$_{\pm 6.7}$ | 57.9$_{\pm 9.7}$ | 66.2$_{\pm 16.7}$ | 82.0$_{\pm 2.1}$ | 72.2$_{\pm 13.9}$ | 53.0$_{\pm 1.7}$ | 54.8$_{\pm 4.2}$ | 68.5$_{\pm 8.1}$ |
| | **kNN-prompt** | 83.0$_{\pm 6.8}$ | 87.3$_{\pm 4.8}$ | 73.4$_{\pm 8.2}$ | 83.8$_{\pm 5.7}$ | 41.4$_{\pm 0.8}$ | 86.5$_{\pm 3.0}$ | 86.4$_{\pm 2.1}$ | 85.2$_{\pm 4.0}$ | 52.3$_{\pm 0.0}$ | 54.7$_{\pm 4.4}$ | 73.4$_{\pm 4.0}$ |
| | **kNN-prompting** | 87.2$_{\pm 2.8}$ | 82.5$_{\pm 1.2}$ | 82.9$_{\pm 4.1}$ | 86.6$_{\pm 2.1}$ | 61.4$_{\pm 4.8}$ | 89.1$_{\pm 1.7}$ | 95.2$_{\pm 1.2}$ | 84.3$_{\pm 1.9}$ | 49.1$_{\pm 3.3}$ | 67.9$_{\pm 5.5}$ | 78.6$_{\pm 2.9}$ |
| | **FADS-ICL** | 88.2$_{\pm 1.8}$ | 90.5$_{\pm 1.7}$ | 83.6$_{\pm 1.5}$ | 87.8$_{\pm 1.0}$ | 78.6$_{\pm 3.6}$ | 91.2$_{\pm 0.8}$ | 98.1$_{\pm 0.7}$ | 87.0$_{\pm 1.3}$ | 58.7$_{\pm 3.7}$ | 84.4$_{\pm 4.0}$ | **84.8**$_{\pm 2.0}$ |
| $m=128$ | **ICL** | 81.3$_{\pm 5.4}$ | 64.1$_{\pm 11.3}$ | 75.2$_{\pm 8.8}$ | 78.2$_{\pm 6.7}$ | 57.9$_{\pm 9.7}$ | 66.2$_{\pm 16.7}$ | 82.0$_{\pm 2.1}$ | 72.2$_{\pm 13.9}$ | 53.0$_{\pm 1.7}$ | 54.8$_{\pm 4.2}$ | 68.5$_{\pm 8.1}$ |
| | **kNN-prompt** | 86.1$_{\pm 3.9}$ | 87.5$_{\pm 3.3}$ | 74.2$_{\pm 4.0}$ | 84.0$_{\pm 5.3}$ | 42.5$_{\pm 1.5}$ | 87.0$_{\pm 3.5}$ | 86.4$_{\pm 2.2}$ | 87.3$_{\pm 2.1}$ | 52.3$_{\pm 0.0}$ | 55.3$_{\pm 4.1}$ | 74.3$_{\pm 3.0}$ |
| | **kNN-prompting** | 86.0$_{\pm 2.8}$ | 83.9$_{\pm 2.2}$ | 82.1$_{\pm 1.6}$ | 87.8$_{\pm 1.5}$ | 65.0$_{\pm 3.0}$ | 88.8$_{\pm 2.2}$ | 96.8$_{\pm 1.0}$ | 86.2$_{\pm 0.8}$ | 51.4$_{\pm 1.8}$ | 74.0$_{\pm 3.2}$ | 80.2$_{\pm 2.0}$ |
| | **FADS-ICL** | 88.9$_{\pm 0.8}$ | 92.1$_{\pm 0.9}$ | 84.8$_{\pm 1.2}$ | 88.4$_{\pm 1.1}$ | 83.2$_{\pm 4.5}$ | 90.5$_{\pm 1.9}$ | 98.6$_{\pm 0.7}$ | 86.1$_{\pm 1.0}$ | 57.8$_{\pm 4.1}$ | 90.5$_{\pm 1.6}$ | **86.1**$_{\pm 1.8}$ |

Table 2: Results under varying data settings. We re-implement kNN-prompt, while kNN-prompting is reproduced using the released code[4]. Here, the LLM scale is 1.5B. Underline indicates leading all baselines with $p < 0.05$.

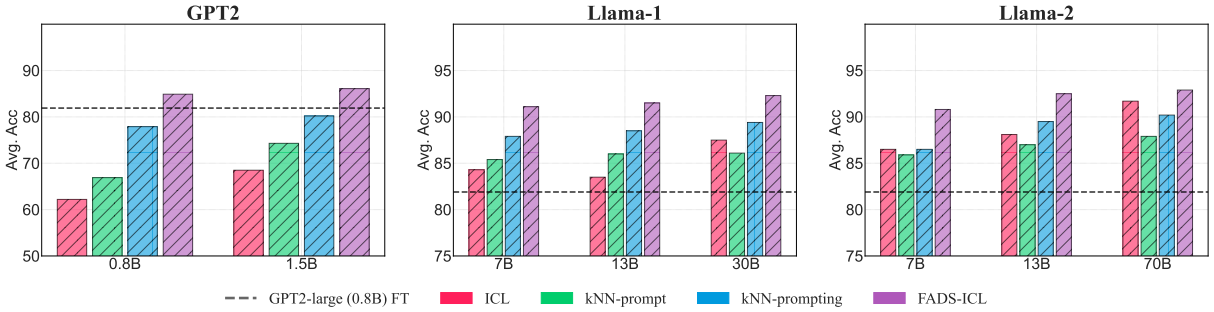


Figure 3: Results across LLM scales with 128-shots.

## 4.2 Main Results

Table 2 and Figure 3 show the results across varying data settings and LLM scale settings[7] respectively. Experimental results show that FADS-ICL consistently outperforms all baselines under all settings by a significant margin, verifying its effectiveness and superiority on downstream tasks.

**For Varying Data Settings** From Table 2, the performance of all methods improves significantly as available samples grow. However, due to data non-scalability vanilla ICL stops improving and keeps 68.5 average accuracy on 10 datasets after training shots reach 32. FADS-ICL can achieve **+14.3** improvement over ICL with 32 shots by feature adaptation, and can further achieve **+17.6** improvement with 128 shots, showing data scalability. Besides, compared to the previous state-of-the-art method, kNN-prompting, FADS-ICL can achieve the improvement of **+6.2** average accuracy with 32 shots, and this large margin roughly keeps as available data scales up. From the perspective of stability, KNN-based methods and FADS-ICL can both gradually lower the standard deviation of performance along with increasing data, while FADS-ICL can consistently obtain better stability than others under fixed data settings.

[7]Full results are listed in Appendix A.

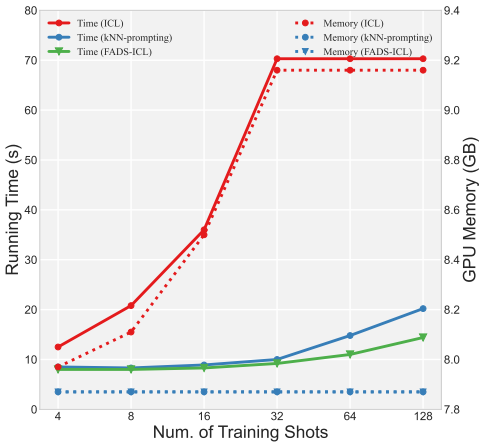


Figure 4: The comparison for computational and memory overhead on the MPQA dataset (256 test samples).

**For Varying LLM Scale Settings** From Figure 3, we can see the performance of all methods achieve consistent improvement roughly as the LLM scale increases along with the capability of the LLM itself, and FADS-ICL still consistently performs much better than all other baselines. Besides, with the increasing context length (the context length of GPT-2, Llama-1, and Llama-2 series is respectively 1k, 2k, and 4k.), the number of demonstrations used in ICL will increase proportionally. Thus, the performance of ICL has been significantly improved and is even comparable to kNN-prompting on the Llama-2 series, which does not benefit from increasing context length, but there is still a considerable gap from FADS-ICL which can refine general features into task-specific features. Surprisingly, under the 0.8B setting, FADS-ICL obtains the best 84.9 average accuracy, even exceeding 81.9 for the entire LLM fine-tuning, which requires fitting the parameters that are 5 orders of magnitude larger than FADS-ICL. It reveals that although fine-tuning is always the best way to adapt to downstream tasks, FADS-ICL effectively fitting extremely few designed parameters, can achieve the best performance at minimal cost in few-shot scenarios.

**Overhead Comparison** Figure 4 shows the comparison for computational and memory overhead of FADS-ICL and baselines. We can see that FADS-ICL is significantly lower than ICL in both running time and GPU memory, especially with more available samples. This is because ICL puts as many demonstrations into context as possible and thus the computational and memory overhead will increase nearly quadratically to compute cross-attention within the LLM as available samples increase, and correspondingly, the input length increases. In comparison, FADS-ICL takes one sample per class as a demonstration all the time while it needs several forward passes of the LLM equal to the number of available samples, which causes the computational overhead to increase linearly and the memory overhead unchanged. Considering both points, the overall computational and memory overhead of FADS-ICL are even smaller than those of vanilla ICL. Note again that the overall overhead (including fitting and inference) of the newly introduced modulator in FADS-ICL is about 0.2s on the CPU, which is negligible compared to the LLM overhead. Besides, kNN-prompting takes similar overheads to FADS-ICL due to the same input form, while it takes a little more running time to compute distances between labeled examples and each test sample rather than directly perform prediction based on task-specific features in FADS-ICL.

## 5 Analysis

In this section, we first systematically analyze the impacts of three key settings in FADS-ICL including modulators, selection of general features, and the number of demonstrations, which provides in-depth insights and the corresponding recommended configurations in FADS-ICL. Visualization for feature adaptation is presented at last.

### 5.1 Effect of Modulators

The modulator contributes to refining features for downstream tasks and performing the final prediction, of which the choice is quite important to FADS-ICL. In this section, we conduct a comparative experiment on parametric modulators and non-parametric modulators, to explore the effect of different modulators in FADS-ICL.

**Setup** For the choice of modulators in FADS-ICL, we consider two groups: parametric modulators and non-parametric modulators. The parametric modulators include *MLP*, *Logistic Regression*, and *Linear SVM*, while the non-parametric modulators involve *Nearest Neighbors* and *Decision Tree*. Almost all hyper-parameters in modulators are set as default values in scikit-learn.

**Result** From Figure 5(a), we can see the parametric modulator group consistently outperforms all other baselines and the non-parametric group by large margins under all data settings, and there are only negligible performance gaps in the paramet-

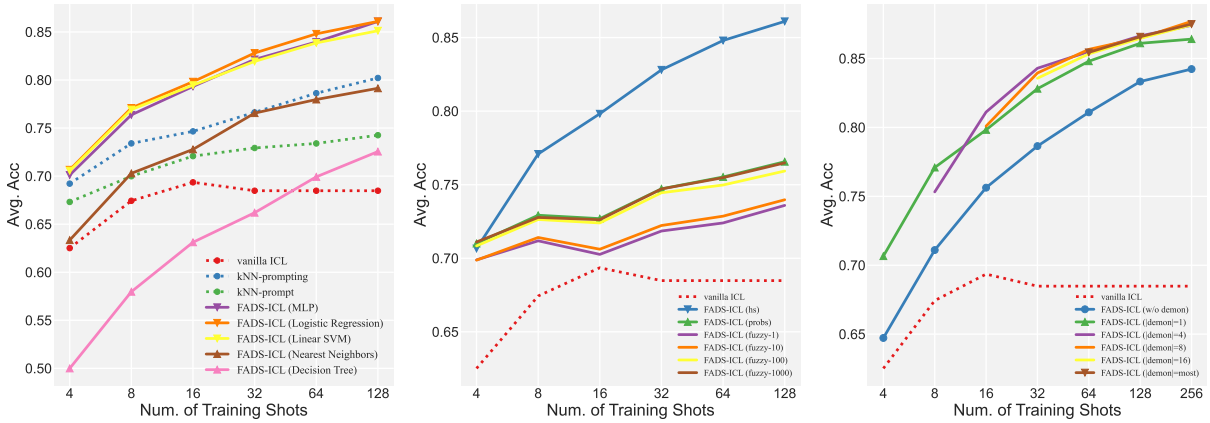


Figure 5: **Left (a):** The effect of different modulators in FADS-ICL. **Middle (b):** The effect of different features as the general features in FADS-ICL. **Right (c):** The role of demonstrations in FADS-ICL.

ric modulator group. This shows the superiority of the parametric modulator in FADS-ICL, which can further refine general features from the LLM and adapt to downstream tasks better. For non-parametric modulators, *Nearest Neighbors* is more superior and stable than *Decision Tree*.

**Analysis and Recommendation** First, we just need to consider parametric modulators, as all three of them exhibit strong performance with only minor differences. Next, considering the number of parameters, *Logistic Regression* and *Linear SVM* are preferred. Further, *Logistic Regression* is more simple and efficient than *Linear SVM*, which needs more training time. In conclusion, the modulator we recommend is *Logistic Regression*.

### 5.2 Effect of Feature Choice

In this subsection, we explore the effect of different choices of general features directly extracted from the LLM in FADS-ICL.

**Setup** For the choice of the general feature in FADS-ICL, we consider two kinds: the last hidden state and probability distribution on vocab. Considering the sparsity of probability distribution, we explore extra settings for probability distribution, *i.e.* fuzzy-k distribution, $k \in \{1, 10, 100, 1000\}$. Specifically, we choose probabilities of the top k most similar tokens to each verbalized label as features used in FADS-ICL. For example, fuzzy-1 probability denotes that only the probabilities of verbalized labels are involved in FADS-ICL.

**Result** From Figure 5(b), we can see the FADS-ICL with the last hidden states as general features consistently outperforms the other group of probability distribution when the training shots surpass 4, and the gap becomes increasingly significant with increasing data. Besides, we observe another consistent phenomenon within the probability distribution group: with more dimensions remaining, performance consistently improves. This suggests that in FADS-ICL, selection for feature dimensions is not necessary, and even dimensions with low information content can be effectively utilized.

**Analysis and Recommendation** In knowledge distillation, Jiao et al. (2020) shows that hidden states contain richer knowledge than probability distributions in general scenarios, while probability distributions can contribute to specific tasks as soft labels. For FADS-ICL, what we need are general features to prepare for subsequent refinement. Therefore, if there are enough labeled samples as supervision, extracting task-adaptive features from hidden states tends to perform much better. Additionally, with few available samples, *e.g.* $m = 4$, choosing probability distributions with simple surface semantic information is slightly more suitable for efficiently fitting model parameters. In short, we recommend the hidden states as the general features in FADS-ICL in most scenarios.

### 5.3 The Role of Demonstrations

In this section, we investigate the role of demonstrations in FADS-ICL, including the impact of whether there is a demonstration and the influence of different numbers of demonstrations.

**Setup** We investigate four settings about demonstrations: (1) *w/o demon* means that no demonstration is prepended to each sample when extracting the general features. (2) *|demon|=1* is the common

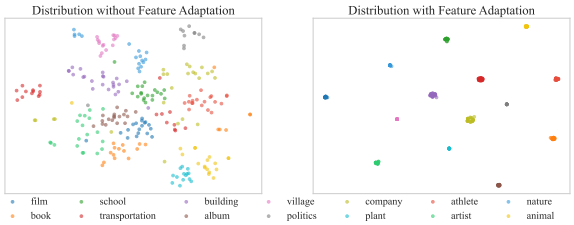


Figure 6: Visualization for feature adaptation.

setting before, *i.e.*, one labeled sample per class is chosen as a demonstration. (3)|*demon*|=$k$ denotes that k labeled samples per class are chosen. We only configure this setting when training shots are not less than $2 * k$ to ensure that there are enough residual labeled samples left for the supervision of the modulator. (4) |*demon*|=*most* means that as many labeled samples as possible are taken as demonstrations up to the context length.

**Result** Figure 5(c) shows that when fully discarding demonstrations, the performance of FADS-ICL (w/o demon) drops dramatically compared with FADS-ICL ($|demon| = 1$), verifying the importance of demonstrations. Besides, when increasing demonstrations to the maximum, it will improve a little further, but not as significantly as before.

**Analysis and Recommendation** Demonstrations serve as the context, which can help the LLM better understand the downstream task and partly regularize general features toward the specific task. From the perspective of tokens, the LLM is designed for language modeling at first, and thus there are many reasonable tokens as candidates, with some task-related and some task-irrelevant. Demonstrations can partly filter task-irrelevant tokens and regularize task-related tokens, which is shown in Appendix C. Equally for features, the role of demonstrations is feature regularization toward the specific task, which is quite beneficial to further refinement of features in FADS-ICL. Finally, we recommend that you take $|demon| = 1$ for simplicity, and try to fill the context with demonstrations when sufficient samples are available.

### 5.4 Visualization for Feature Adaptation

We conduct visualization of feature distribution without/with feature adaptation on the DBPedia dataset. From Figure 6, we can see that the general features from the LLM maintain a certain degree of discrimination for downstream tasks, but the discrimination of features refined to specific tasks is much higher than that of general features. Therefore, it is essential to refine general features in the LLM to task-specific features to perform the downstream task, that is feature adaptation.

## 6 Related Works

There are many works contributing to improving the performance of ICL on downstream tasks. First, most related works focus on designing the best prefix for each test sample, including demonstration selection, corresponding order, and task-specific templates. Specifically, Liu et al. (2022) retrieves demonstration sets based on semantic similarity with an off-shelf sentence encoder, while Rubin et al. (2022); Wang et al. (2023) train a prompt retriever with weak supervised learning. Some works (Lu et al., 2022a; Wu et al., 2023) explore the order of demonstrations, while other works (Sorensen et al., 2022; Prasad et al., 2023) aim to search for the most suitable templates for specific tasks. These works do not conflict with FADS-ICL, and you can expect better results when these methods are applied to our framework. Second, a few works further finetune the LLM based on the input-output format of ICL with a large set of tasks, to adapt to ICL during inference, *e.g.* MetaICL (Min et al., 2022), ICT (Chen et al., 2022b). This type of approach has not become a mainstream application paradigm due to the high cost of finetuning LLM. Third, some works (Shi et al., 2022; Xu et al., 2023) have explored exploiting beyond-context samples, similar to KNN-LM (Khandelwal et al., 2020). However, they ignore feature refinement for specific tasks and instead directly use features designed for language modeling in LLM, while FADS-ICL implements feature adaptation, resulting in better performance on downstream tasks.

## 7 Conclusion

This paper proposes FADS-ICL, which can leverage task-adaptive features to promote inference on the downstream task with the supervision of beyond-context samples. It implements data scalability by extracting general features using the LLM with ICL input form one by one, while a lightweight task-specific modulator is introduced to complete feature refinement and final prediction, achieving feature adaptation. FADS-ICL consistently outperforms previous state-of-the-art by a significant margin under all data and LLM scale settings, verifying its effectiveness and superior-

ity. Detailed analytical experiments on key settings provide more in-depth insights and recommended configurations in FADS-ICL.

## Limitation

The main limitation of FADS-ICL is that it can be easily adapted to any downstream classification tasks, but is difficult to apply to natural language generation tasks. As usually there are only a few or a dozen categories in classification tasks, it is easy to collate several annotated samples for each class. However, natural language generation tasks can be regarded as classification tasks on the entire vocab with tens of thousands of tokens, thus the cost of data annotation becomes unacceptably heavy. Besides, the number of parameters in the modulator is also related to the number of categories and its increase cannot be underestimated either.

## Acknowledgements

We would like to thank all the reviewers for their valuable suggestions, which significantly improved this paper. This research is supported by National Science Fund for Excellent Young Scholars under Grant 62222212 and the General Program of National Natural Science Foundation of China under Grant 62376033.

## References

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*.

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. 2013. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122.

Guanzheng Chen, Fangyu Liu, Zaiqiao Meng, and Shangsong Liang. 2022a. Revisiting parameter-efficient tuning: Are we really there yet? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 2612–2626. Association for Computational Linguistics.

Yanda Chen, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. 2022b. Meta-learning via language model in-context tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 719–730. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges, Evaluating Predictive Uncertainty, Visual Object Classification and Recognizing Textual Entailment, First PASCAL Machine Learning Challenges Workshop, MLCW 2005, Southampton, UK, April 11-13, 2005, Revised Selected Papers*, volume 3944 of *Lecture Notes in Computer Science*, pages 177–190. Springer.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 168–177. ACM.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4163–4174. Association for Computational Linguistics.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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: The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, DeeLIO@ACL 2022, Dublin, Ireland and Online, May 27, 2022*, pages 100–114. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022a. 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), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8086–8098. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022b. 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.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022. 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, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 2791–2809. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain*, pages 271–278. ACL.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, pages 115–124. The Association for Computer Linguistics.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2023. Grips: Gradient-free, edit-based instruction search for prompting large language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 3827–3846. Association for Computational Linguistics.

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, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 2655–2671. Association for Computational Linguistics.

Weijia Shi, Julian Michael, Suchin Gururangan, and Luke Zettlemoyer. 2022. Nearest neighbor zero-shot inference. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 3254–3265. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. 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, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Taylor Sorensen, Joshua Robinson, Christopher Michael Rytting, Alexander Glenn Shaw, Kyle Jeffrey Rogers, Alexia Pauline 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), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 819–862. Association for Computational Linguistics.

Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. In *SIGIR 2000: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 24-28, 2000, Athens, Greece*, pages 200–207. ACM.

Liang Wang, Nan Yang, and Furu Wei. 2023. Learning to retrieve in-context examples for large language models. *CoRR*, abs/2307.07164.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Lang. Resour. Evaluation*, 39(2-3):165–210.

Zhiyong Wu, Yaoxiang Wang, Jiacheng Ye, and Lingpeng Kong. 2023. Self-adaptive in-context learning: An information compression perspective for in-context example selection and ordering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 1423–1436. Association for Computational Linguistics.

Benfeng Xu, Quan Wang, Zhendong Mao, Yajuan Lyu, Qiaoqiao She, and Yongdong Zhang. 2023. $k$nn prompting: Beyond-context learning with calibration-free nearest neighbor inference. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

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.

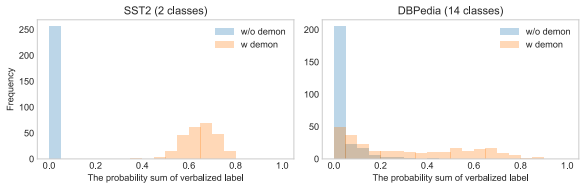


Figure 7: The impact of the demonstrations on the probability distribution of verbalized labels in vanilla ICL.

## A Performance Across LLM Scales

Here, We provide detailed experimental results across LLM scales in Table 3.

## B Comparison to Baselines based on Demonstration Selection

In this subsection, We compare FADS-ICL with KATE (Liu et al., 2022) based on demonstration selection via similarity measure. Specifically, we consider multiple similarity measurement methods and compare experimental results when available samples are beyond the context. The full result is presented in Table 6. We can see that KATEs based on all similarity measurement methods have provided a slight performance improvement, which is completely incomparable with FADS-ICL.

## C The Role of Demonstrations in ICL

Figure 7 presents the distributions of the probability sum of task-specific labels without/with demonstrations in vanilla ICL. It shows that the probability sum of task-specific labels is quite low without demonstrations, and demonstrations improve it significantly. This shows that demonstrations can partly filter task-irrelevant tokens and regularize task-related tokens.

## D Black-box LLMs

In the main experiment, the effectiveness of the method has been verified based on various open-source LLMs (GPT-2, Llama-1, and Llama-2 series). For those black-box LLMs mainly used for chat (i.e. text generation), our method is indeed not applicable because the features that we need in the LLM cannot be obtained through the API. Fortunately, most AI companies in the black-box LLM business have a dedicated business branch to provide feature output for users, such as text-embedding-ada-002[8] released by OpenAI, text-embedding-v2[9] released by Alibaba Cloud, etc. We experiment on OpenAI 's text-embedding-ada-002. Since this black-box LLM does not provide the text generation function, vanilla ICL is unavailable. We give the previous SOTA method, kNN-prompting, as a comparison and the experimental result is present in Table 4. It shows that the proposed method provides significant improvements compared to kNN-prompting, which is consistent with the conclusion in the main experiment.

## E FADS-ICL vs PEFT

We provide below a comparison of FADS-ICL, fine-tuning, and parameter-efficient fine-tuning (PEFT) under various data settings. Experiments prove that fine-tuning and PEFT are not suitable for low-resource scenarios.

As can be seen from Table 5, in the case of low resources (such as 32, 64-shot), the performance of the fine-tuning method is far inferior to KNN-prompting and FADS-ICL based on in-context learning because training data is far from sufficient to support transfer learning from the original language modeling task to downstream tasks. When the available data grows to 128-shot, the fine-tuning method improves rapidly and becomes competitive, but there is still a certain gap from FADS-ICL.

In addition, PEFT, such as LoRA, is also not suitable for low-resource scenarios due to their high instability. It can be seen from the overall results that although Lora reduces the training parameters to a certain extent, the average performance is far inferior to the fine-tuning method and FADS-ICL. Careful inspection revealed that due to the small amount of training data, Lora's training was extremely unstable and could easily lead to model collapse. For example, in five random runs on the SST2 dataset, the highest accuracy can reach 93.8 while the lowest is only 64.5 with only different seeds, and training on CR resulted in model collapse with performance approaching random guessing. As pointed out in previous work (Chen et al., 2022a), PEFT (including Adapter, Prompt tuning, LoRA, and BitFit) exhibit high instability in low resource settings subject to newly introduced parameter initialization and training data order, determined by random seeds.

Compared with them, FADS-ICL has the follow-

[8] https://openai.com/blog/new-and-improved-embedding-model

[9] https://help.aliyun.com/zh/dashscope/developer-reference/generic-text-vector

| Models&Methods | | SST2 | SUBJ | MPQA | AGNews | CB | CR | DBPedia | MR | RTE | TREC | AVG |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **BERT Large FT** | | 88.3$_{\pm 1.4}$ | 90.7$_{\pm 0.6}$ | 74.5$_{\pm 5.0}$ | 88.0$_{\pm 1.0}$ | 78.6$_{\pm 3.6}$ | 88.0$_{\pm 2.6}$ | 95.1$_{\pm 1.8}$ | 83.0$_{\pm 3.1}$ | 58.1$_{\pm 1.5}$ | 78.8$_{\pm 5.3}$ | 82.3$_{\pm 2.6}$ |
| **GPT Large FT** | | 90.7$_{\pm 1.3}$ | 86.1$_{\pm 1.7}$ | 87.6$_{\pm 0.9}$ | 88.3$_{\pm 1.5}$ | 70.0$_{\pm 2.0}$ | 86.7$_{\pm 13.2}$ | 96.5$_{\pm 1.2}$ | 86.2$_{\pm 1.0}$ | 55.4$_{\pm 3.8}$ | 71.2$_{\pm 2.2}$ | 81.9$_{\pm 2.9}$ |
| **0.8B** | **ICL** | 63.4$_{\pm 7.3}$ | 58.9$_{\pm 8.7}$ | 70.5$_{\pm 5.2}$ | 60.7$_{\pm 12.2}$ | 37.1$_{\pm 14.5}$ | 83.3$_{\pm 13.7}$ | 63.4$_{\pm 6.0}$ | 77.0$_{\pm 15.7}$ | 53.6$_{\pm 3.1}$ | 54.2$_{\pm 8.6}$ | 62.2$_{\pm 9.5}$ |
| | **kNN-prompt** | 88.0$_{\pm 1.2}$ | 73.0$_{\pm 7.8}$ | 76.6$_{\pm 3.0}$ | 51.4$_{\pm 22.4}$ | 41.4$_{\pm 0.8}$ | 86.0$_{\pm 2.5}$ | 68.0$_{\pm 4.0}$ | 85.0$_{\pm 3.0}$ | 52.3$_{\pm 0.0}$ | 47.7$_{\pm 7.1}$ | 66.9$_{\pm 5.2}$ |
| | **kNN-prompting** | 83.2$_{\pm 2.8}$ | 79.8$_{\pm 1.8}$ | 72.3$_{\pm 7.4}$ | 86.9$_{\pm 1.8}$ | 67.5$_{\pm 7.1}$ | 86.6$_{\pm 2.6}$ | 95.5$_{\pm 0.7}$ | 82.4$_{\pm 2.9}$ | 49.5$_{\pm 1.8}$ | 75.3$_{\pm 3.8}$ | 77.9$_{\pm 3.3}$ |
| | **FADS-ICL** | 87.7$_{\pm 2.4}$ | 90.1$_{\pm 1.9}$ | 81.7$_{\pm 1.0}$ | 88.1$_{\pm 1.4}$ | 80.0$_{\pm 2.3}$ | 88.7$_{\pm 1.6}$ | 97.7$_{\pm 0.6}$ | 85.0$_{\pm 2.2}$ | 59.4$_{\pm 1.9}$ | 90.3$_{\pm 1.5}$ | **84.9**$_{\pm \mathbf{1.7}}$ |
| **1.5B** | **ICL** | 81.3$_{\pm 5.4}$ | 64.1$_{\pm 11.3}$ | 75.2$_{\pm 8.8}$ | 78.2$_{\pm 6.7}$ | 57.9$_{\pm 9.7}$ | 66.2$_{\pm 16.7}$ | 82.0$_{\pm 2.1}$ | 72.2$_{\pm 13.9}$ | 53.0$_{\pm 1.7}$ | 54.8$_{\pm 4.2}$ | 68.5$_{\pm 8.1}$ |
| | **kNN-prompt** | 86.1$_{\pm 3.9}$ | 87.5$_{\pm 3.3}$ | 74.2$_{\pm 4.0}$ | 84.0$_{\pm 5.3}$ | 42.5$_{\pm 1.5}$ | 87.0$_{\pm 3.5}$ | 86.4$_{\pm 2.2}$ | 87.3$_{\pm 2.1}$ | 52.3$_{\pm 0.0}$ | 55.3$_{\pm 4.1}$ | 74.3$_{\pm 3.0}$ |
| | **kNN-prompting** | 86.0$_{\pm 2.8}$ | 83.9$_{\pm 2.2}$ | 82.1$_{\pm 1.6}$ | 87.8$_{\pm 1.5}$ | 65.0$_{\pm 3.0}$ | 88.8$_{\pm 2.2}$ | 96.8$_{\pm 1.0}$ | 86.2$_{\pm 0.8}$ | 51.4$_{\pm 1.8}$ | 74.0$_{\pm 3.2}$ | 80.2$_{\pm 2.0}$ |
| | **FADS-ICL** | 88.9$_{\pm 0.8}$ | 92.1$_{\pm 0.9}$ | 84.8$_{\pm 1.2}$ | 88.4$_{\pm 1.1}$ | 83.2$_{\pm 4.5}$ | 90.5$_{\pm 1.9}$ | 98.6$_{\pm 0.7}$ | 86.1$_{\pm 1.0}$ | 57.8$_{\pm 4.1}$ | 90.5$_{\pm 1.6}$ | **86.1**$_{\pm \mathbf{1.8}}$ |
| **7B** | **ICL** | 93.3$_{\pm 0.7}$ | 79.6$_{\pm 14.5}$ | 86.8$_{\pm 1.4}$ | 86.2$_{\pm 0.9}$ | 74.6$_{\pm 11.3}$ | 89.8$_{\pm 2.4}$ | 81.2$_{\pm 1.5}$ | 93.3$_{\pm 1.2}$ | 74.1$_{\pm 1.6}$ | 83.9$_{\pm 3.9}$ | 84.3$_{\pm 3.9}$ |
| | **kNN-prompt** | 93.3$_{\pm 0.8}$ | 82.8$_{\pm 15.3}$ | 83.6$_{\pm 1.8}$ | 86.4$_{\pm 2.2}$ | 80.4$_{\pm 1.8}$ | 92.3$_{\pm 1.7}$ | 91.6$_{\pm 1.1}$ | 92.7$_{\pm 0.3}$ | 69.8$_{\pm 3.5}$ | 81.1$_{\pm 6.0}$ | 85.4$_{\pm 3.4}$ |
| | **kNN-prompting** | 92.8$_{\pm 0.8}$ | 93.4$_{\pm 1.4}$ | 84.6$_{\pm 1.3}$ | 87.6$_{\pm 1.4}$ | 84.3$_{\pm 2.3}$ | 91.9$_{\pm 2.1}$ | 99.0$_{\pm 0.3}$ | 91.8$_{\pm 0.4}$ | 70.4$_{\pm 4.7}$ | 83.4$_{\pm 3.8}$ | 87.9$_{\pm 1.8}$ |
| | **FADS-ICL** | 92.4$_{\pm 1.0}$ | 95.6$_{\pm 0.3}$ | 85.8$_{\pm 1.7}$ | 89.7$_{\pm 0.7}$ | 95.0$_{\pm 2.3}$ | 93.2$_{\pm 1.1}$ | 98.9$_{\pm 0.4}$ | 92.4$_{\pm 0.6}$ | 74.7$_{\pm 0.9}$ | 93.6$_{\pm 1.8}$ | **91.1**$_{\pm \mathbf{1.1}}$ |
| **7B$^2$** | **ICL** | 93.6$_{\pm 1.5}$ | 90.2$_{\pm 1.5}$ | 87.0$_{\pm 1.8}$ | 85.5$_{\pm 1.6}$ | 70.0$_{\pm 7.0}$ | 90.4$_{\pm 2.4}$ | 95.7$_{\pm 0.9}$ | 92.9$_{\pm 1.0}$ | 74.0$_{\pm 2.1}$ | 85.5$_{\pm 2.0}$ | 86.5$_{\pm 2.2}$ |
| | **kNN-prompt** | 94.5$_{\pm 1.0}$ | 79.5$_{\pm 18.1}$ | 81.7$_{\pm 3.4}$ | 85.9$_{\pm 1.7}$ | 80.0$_{\pm 4.6}$ | 93.4$_{\pm 1.0}$ | 98.3$_{\pm 0.5}$ | 93.2$_{\pm 1.1}$ | 68.4$_{\pm 4.6}$ | 84.1$_{\pm 4.9}$ | 85.9$_{\pm 4.1}$ |
| | **kNN-prompting** | 92.8$_{\pm 1.6}$ | 95.5$_{\pm 0.6}$ | 81.7$_{\pm 2.1}$ | 88.3$_{\pm 1.7}$ | 77.1$_{\pm 4.6}$ | 93.7$_{\pm 0.7}$ | 99.0$_{\pm 0.2}$ | 92.2$_{\pm 0.6}$ | 69.1$_{\pm 3.0}$ | 85.1$_{\pm 2.4}$ | 86.5$_{\pm 1.8}$ |
| | **FADS-ICL** | 92.2$_{\pm 1.0}$ | 95.2$_{\pm 0.8}$ | 87.3$_{\pm 2.1}$ | 89.1$_{\pm 0.7}$ | 90.7$_{\pm 3.2}$ | 93.4$_{\pm 1.2}$ | 99.1$_{\pm 0.4}$ | 92.2$_{\pm 0.7}$ | 75.8$_{\pm 3.1}$ | 93.0$_{\pm 1.4}$ | **90.8**$_{\pm \mathbf{1.5}}$ |
| **13B** | **ICL** | 94.9$_{\pm 0.9}$ | 88.6$_{\pm 2.8}$ | 85.5$_{\pm 2.0}$ | 86.2$_{\pm 1.9}$ | 62.5$_{\pm 6.8}$ | 87.8$_{\pm 2.7}$ | 81.5$_{\pm 1.0}$ | 93.8$_{\pm 0.3}$ | 71.2$_{\pm 5.1}$ | 83.4$_{\pm 2.6}$ | 83.5$_{\pm 2.6}$ |
| | **kNN-prompt** | 93.9$_{\pm 1.0}$ | 88.3$_{\pm 6.1}$ | 84.3$_{\pm 2.2}$ | 89.9$_{\pm 1.0}$ | 80.0$_{\pm 3.4}$ | 91.5$_{\pm 0.6}$ | 95.1$_{\pm 2.9}$ | 92.7$_{\pm 0.9}$ | 68.3$_{\pm 3.8}$ | 75.8$_{\pm 8.3}$ | 86.0$_{\pm 3.0}$ |
| | **kNN-prompting** | 93.5$_{\pm 1.1}$ | 92.2$_{\pm 1.7}$ | 84.4$_{\pm 0.7}$ | 88.4$_{\pm 1.1}$ | 88.9$_{\pm 4.3}$ | 92.3$_{\pm 1.7}$ | 99.1$_{\pm 0.3}$ | 93.1$_{\pm 0.6}$ | 68.4$_{\pm 1.7}$ | 84.8$_{\pm 2.0}$ | 88.5$_{\pm 1.5}$ |
| | **FADS-ICL** | 93.1$_{\pm 0.3}$ | 95.7$_{\pm 1.1}$ | 85.9$_{\pm 1.1}$ | 89.2$_{\pm 0.4}$ | 97.1$_{\pm 1.0}$ | 92.3$_{\pm 0.8}$ | 99.0$_{\pm 0.3}$ | 93.4$_{\pm 0.9}$ | 75.0$_{\pm 3.2}$ | 93.8$_{\pm 1.4}$ | **91.5**$_{\pm \mathbf{1.1}}$ |
| **13B$^2$** | **ICL** | 94.9$_{\pm 0.9}$ | 88.6$_{\pm 2.8}$ | 85.5$_{\pm 2.0}$ | 86.2$_{\pm 1.9}$ | 62.5$_{\pm 6.8}$ | 87.8$_{\pm 2.7}$ | 81.5$_{\pm 1.0}$ | 93.8$_{\pm 0.3}$ | 71.2$_{\pm 5.1}$ | 83.4$_{\pm 2.6}$ | 88.1$_{\pm 2.1}$ |
| | **kNN-prompt** | 93.1$_{\pm 1.0}$ | 81.0$_{\pm 15.9}$ | 85.5$_{\pm 1.9}$ | 87.9$_{\pm 1.4}$ | 86.1$_{\pm 2.3}$ | 94.1$_{\pm 0.9}$ | 97.3$_{\pm 1.3}$ | 92.7$_{\pm 0.8}$ | 74.8$_{\pm 4.3}$ | 77.4$_{\pm 6.3}$ | 87.0$_{\pm 3.6}$ |
| | **kNN-prompting** | 94.6$_{\pm 0.9}$ | 94.6$_{\pm 1.2}$ | 84.5$_{\pm 1.6}$ | 87.1$_{\pm 2.2}$ | 91.4$_{\pm 1.5}$ | 92.2$_{\pm 2.5}$ | 98.8$_{\pm 0.2}$ | 92.6$_{\pm 0.9}$ | 72.8$_{\pm 3.7}$ | 86.7$_{\pm 2.3}$ | 89.5$_{\pm 1.7}$ |
| | **FADS-ICL** | 93.1$_{\pm 1.0}$ | 96.2$_{\pm 0.8}$ | 88.0$_{\pm 0.8}$ | 89.6$_{\pm 0.7}$ | 96.1$_{\pm 2.0}$ | 93.8$_{\pm 0.9}$ | 98.8$_{\pm 0.3}$ | 93.4$_{\pm 0.3}$ | 81.1$_{\pm 3.0}$ | 95.0$_{\pm 0.8}$ | **92.5**$_{\pm \mathbf{1.1}}$ |
| **30B** | **ICL** | 94.0$_{\pm 0.3}$ | 91.2$_{\pm 2.5}$ | 87.6$_{\pm 1.1}$ | 87.1$_{\pm 1.8}$ | 88.2$_{\pm 3.7}$ | 89.1$_{\pm 2.0}$ | 81.2$_{\pm 0.5}$ | 94.7$_{\pm 0.7}$ | 77.2$_{\pm 3.0}$ | 84.6$_{\pm 1.7}$ | 87.5$_{\pm 1.7}$ |
| | **kNN-prompt** | 93.9$_{\pm 1.0}$ | 89.6$_{\pm 4.4}$ | 84.3$_{\pm 2.2}$ | 89.9$_{\pm 1.0}$ | 80.0$_{\pm 3.4}$ | 91.5$_{\pm 0.6}$ | 95.1$_{\pm 2.9}$ | 92.7$_{\pm 0.9}$ | 68.3$_{\pm 3.8}$ | 75.8$_{\pm 8.3}$ | 86.1$_{\pm 2.9}$ |
| | **kNN-prompting** | 92.8$_{\pm 0.7}$ | 93.6$_{\pm 1.3}$ | 84.3$_{\pm 1.6}$ | 89.0$_{\pm 1.6}$ | 88.9$_{\pm 2.3}$ | 92.6$_{\pm 0.8}$ | 98.8$_{\pm 0.3}$ | 92.7$_{\pm 0.4}$ | 74.8$_{\pm 2.3}$ | 86.6$_{\pm 2.1}$ | 89.4$_{\pm 1.3}$ |
| | **FADS-ICL** | 92.7$_{\pm 0.6}$ | 96.4$_{\pm 0.7}$ | 87.3$_{\pm 1.6}$ | 90.5$_{\pm 0.7}$ | 94.6$_{\pm 0.0}$ | 91.9$_{\pm 1.7}$ | 98.9$_{\pm 0.3}$ | 93.8$_{\pm 1.6}$ | 81.8$_{\pm 2.0}$ | 95.2$_{\pm 0.8}$ | **92.3**$_{\pm \mathbf{1.0}}$ |
| **70B$^2$** | **ICL** | 93.2$_{\pm 1.6}$ | 96.5$_{\pm 1.1}$ | 89.1$_{\pm 1.3}$ | 88.8$_{\pm 1.3}$ | 95.4$_{\pm 2.7}$ | 92.3$_{\pm 1.9}$ | 94.1$_{\pm 1.6}$ | 93.9$_{\pm 0.4}$ | 85.5$_{\pm 0.3}$ | 88.2$_{\pm 3.9}$ | 91.7$_{\pm 1.6}$ |
| | **kNN-prompt** | 92.0$_{\pm 1.4}$ | 86.0$_{\pm 5.2}$ | 84.3$_{\pm 1.6}$ | 87.5$_{\pm 1.6}$ | 87.9$_{\pm 3.2}$ | 92.5$_{\pm 1.4}$ | 89.5$_{\pm 3.1}$ | 92.4$_{\pm 1.3}$ | 79.4$_{\pm 2.7}$ | 87.6$_{\pm 2.6}$ | 87.9$_{\pm 2.4}$ |
| | **kNN-prompting** | 91.9$_{\pm 2.2}$ | 91.3$_{\pm 3.6}$ | 84.1$_{\pm 1.5}$ | 87.7$_{\pm 1.4}$ | 94.3$_{\pm 0.8}$ | 92.0$_{\pm 1.4}$ | 98.8$_{\pm 0.4}$ | 91.6$_{\pm 1.3}$ | 80.2$_{\pm 2.9}$ | 90.1$_{\pm 2.6}$ | 90.2$_{\pm 1.8}$ |
| | **FADS-ICL** | 92.6$_{\pm 1.1}$ | 96.3$_{\pm 0.7}$ | 87.7$_{\pm 2.1}$ | 90.2$_{\pm 1.0}$ | 98.2$_{\pm 1.3}$ | 93.3$_{\pm 1.6}$ | 98.9$_{\pm 0.3}$ | 93.0$_{\pm 0.5}$ | 83.6$_{\pm 1.7}$ | 95.2$_{\pm 1.5}$ | **92.9**$_{\pm \mathbf{1.2}}$ |

Table 3: Results across LLM scales with 128-shots. Here, 0.8B and 1.5B denote the GPT-2 series; 7B, 13B, and 30B denote the Llama-1 series; 7B$^2$, 13B$^2$ and 70B$^2$ denotes the Llama-2 series.

| **Methods** | **8-shots** | **32-shots** | **128-shots** |
|---|---|---|---|
| **kNN-prompting** | 61.4 | 68.4 | 72.9 |
| **FADS-ICL** | **73.0** | **80.8** | **87.0** |

Table 4: The performance of FADS-ICL on the black-box LLM. The experiment is conducted on OpenAI's text-embedding-ada-002 model.

| **Methods** | **32-shots** | **64-shots** | **128-shots** |
|---|---|---|---|
| **kNN-prompting** | 76.4 | 78.7 | 79.7 |
| **Fine Tuning (0.8B)** | 74.7 | 75.8 | 81.9 |
| **LoRA (1.5M)** | 71.4 | 73.0 | 77.1 |
| **FADS-ICL (2.6K)** | **81.7** | **83.7** | **84.9** |

Table 5: The comparison of FADS-ICL, fine-tuning and parameter-efficient fine-tuning. The experiment is conducted on GPT2-Large and inside the parentheses are the number of parameters involved in training.

ing advantages in low-resource scenarios: much better performance, higher stability, and negligible training overhead (within 1s CPU).

## F Prompt Template

The used templates are presented in Table 7 (adopted from Lu et al. (2022b)), which are intuitively designed.

| Setting & Methods | | SST2 | SUBJ | MPQA | AGNews | CB | CR | DBPedia | MR | RTE | TREC | AVG |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **In-Context Learning** | | $81.3_{\pm 5.4}$ | $64.1_{\pm 11.3}$ | $75.2_{\pm 8.8}$ | $72.7_{\pm 18.5}$ | $60.7_{\pm 2.8}$ | $66.2_{\pm 16.7}$ | $83.5_{\pm 3.8}$ | $72.2_{\pm 13.9}$ | $53.0_{\pm 1.7}$ | $54.2_{\pm 4.9}$ | 68.3 |
| $m = 32$ | **BM25** | $68.4_{\pm 3.7}$ | $63.6_{\pm 3.0}$ | $75.2_{\pm 8.8}$ | $69.0_{\pm 3.6}$ | $65.0_{\pm 6.5}$ | $55.2_{\pm 1.0}$ | $80.7_{\pm 1.3}$ | $59.4_{\pm 1.4}$ | $53.0_{\pm 2.3}$ | $65.5_{\pm 2.8}$ | 65.5 |
| | **SBERT** | $71.0_{\pm 4.9}$ | $67.5_{\pm 1.6}$ | $75.2_{\pm 8.8}$ | $82.3_{\pm 2.1}$ | $62.9_{\pm 2.9}$ | $57.8_{\pm 1.9}$ | $83.8_{\pm 1.2}$ | $57.7_{\pm 4.3}$ | $51.2_{\pm 3.1}$ | $58.4_{\pm 6.7}$ | 66.8 |
| | **SimCSE** | $68.1_{\pm 4.5}$ | $69.8_{\pm 3.1}$ | $75.2_{\pm 8.8}$ | $80.7_{\pm 2.9}$ | $66.4_{\pm 2.3}$ | $55.9_{\pm 2.8}$ | $82.3_{\pm 1.9}$ | $57.3_{\pm 2.4}$ | $52.8_{\pm 1.7}$ | $54.5_{\pm 1.8}$ | 66.3 |
| | **Trans-Encoder** | $67.9_{\pm 3.8}$ | $70.9_{\pm 4.1}$ | $75.2_{\pm 8.8}$ | $77.7_{\pm 2.5}$ | $61.4_{\pm 6.0}$ | $56.7_{\pm 1.7}$ | $82.8_{\pm 2.0}$ | $57.3_{\pm 4.3}$ | $52.7_{\pm 2.0}$ | $59.3_{\pm 3.9}$ | 66.2 |
| | **FADS-ICL** | $88.0_{\pm 4.6}$ | $90.1_{\pm 2.2}$ | $81.4_{\pm 3.0}$ | $84.9_{\pm 1.4}$ | $72.1_{\pm 2.0}$ | $92.3_{\pm 0.7}$ | $97.9_{\pm 0.9}$ | $85.9_{\pm 2.8}$ | $55.5_{\pm 2.9}$ | $79.9_{\pm 4.7}$ | **82.8** |
| $m = 64$ | **BM25** | $69.7_{\pm 2.9}$ | $67.7_{\pm 2.5}$ | $79.4_{\pm 2.0}$ | $71.7_{\pm 1.9}$ | $68.6_{\pm 4.1}$ | $54.7_{\pm 1.1}$ | $83.4_{\pm 2.0}$ | $58.7_{\pm 1.5}$ | $52.0_{\pm 1.6}$ | $65.9_{\pm 5.6}$ | 67.2 |
| | **SBERT** | $71.8_{\pm 3.4}$ | $71.6_{\pm 3.8}$ | $80.2_{\pm 1.5}$ | $84.6_{\pm 2.2}$ | $66.8_{\pm 6.8}$ | $59.8_{\pm 1.8}$ | $84.6_{\pm 0.8}$ | $57.6_{\pm 1.6}$ | $52.8_{\pm 3.6}$ | $63.7_{\pm 4.7}$ | 69.4 |
| | **SimCSE** | $68.4_{\pm 4.7}$ | $69.7_{\pm 2.6}$ | $81.6_{\pm 0.6}$ | $83.1_{\pm 2.2}$ | $71.4_{\pm 4.6}$ | $57.9_{\pm 2.6}$ | $84.8_{\pm 2.1}$ | $57.3_{\pm 2.0}$ | $52.3_{\pm 3.6}$ | $58.1_{\pm 1.8}$ | 68.5 |
| | **Trans-Encoder** | $69.3_{\pm 4.3}$ | $73.0_{\pm 1.2}$ | $82.0_{\pm 2.2}$ | $79.3_{\pm 2.0}$ | $69.3_{\pm 3.4}$ | $57.9_{\pm 2.7}$ | $85.6_{\pm 1.2}$ | $57.9_{\pm 1.4}$ | $52.1_{\pm 4.5}$ | $60.1_{\pm 2.6}$ | 68.7 |
| | **FADS-ICL** | $88.2_{\pm 1.8}$ | $90.5_{\pm 1.7}$ | $83.6_{\pm 1.5}$ | $87.8_{\pm 1.0}$ | $78.6_{\pm 3.6}$ | $91.2_{\pm 0.8}$ | $98.1_{\pm 0.7}$ | $87.0_{\pm 1.3}$ | $58.7_{\pm 3.7}$ | $84.4_{\pm 4.0}$ | **84.8** |
| $m = 128$ | **BM25** | $69.1_{\pm 0.5}$ | $66.8_{\pm 2.7}$ | $75.2_{\pm 6.2}$ | $77.5_{\pm 1.4}$ | $71.4_{\pm 0.0}$ | $56.4_{\pm 1.2}$ | $85.5_{\pm 1.7}$ | $59.8_{\pm 2.0}$ | $54.5_{\pm 1.3}$ | $72.3_{\pm 4.9}$ | 68.9 |
| | **SBERT** | $71.7_{\pm 1.9}$ | $71.9_{\pm 1.7}$ | $79.6_{\pm 3.6}$ | $85.3_{\pm 2.0}$ | $69.6_{\pm 0.0}$ | $58.8_{\pm 0.8}$ | $87.2_{\pm 0.9}$ | $60.1_{\pm 2.3}$ | $53.3_{\pm 2.3}$ | $64.9_{\pm 2.1}$ | 70.2 |
| | **SimCSE** | $70.9_{\pm 2.2}$ | $71.6_{\pm 3.2}$ | $81.6_{\pm 2.6}$ | $84.5_{\pm 1.7}$ | $73.2_{\pm 0.0}$ | $58.5_{\pm 2.2}$ | $87.0_{\pm 2.1}$ | $58.7_{\pm 1.4}$ | $53.4_{\pm 3.2}$ | $59.6_{\pm 3.3}$ | 69.9 |
| | **Trans-Encoder** | $69.0_{\pm 1.3}$ | $75.5_{\pm 2.2}$ | $82.6_{\pm 1.9}$ | $82.9_{\pm 0.6}$ | $73.2_{\pm 0.0}$ | $56.1_{\pm 0.8}$ | $87.0_{\pm 1.4}$ | $57.1_{\pm 1.3}$ | $52.9_{\pm 3.1}$ | $63.9_{\pm 2.3}$ | 70.0 |
| | **FADS-ICL** | $88.9_{\pm 0.8}$ | $92.1_{\pm 0.9}$ | $84.8_{\pm 1.2}$ | $88.4_{\pm 1.1}$ | $83.2_{\pm 4.5}$ | $90.5_{\pm 1.9}$ | $98.6_{\pm 0.7}$ | $86.1_{\pm 1.0}$ | $57.8_{\pm 4.1}$ | $90.5_{\pm 1.6}$ | **86.1** |
| $m = 256$ | **BM25** | $72.0_{\pm 3.9}$ | $72.3_{\pm 1.4}$ | $78.8_{\pm 3.2}$ | $77.3_{\pm 3.0}$ | $71.4_{\pm 0.0}$ | $57.1_{\pm 0.8}$ | $88.2_{\pm 1.9}$ | $58.4_{\pm 1.6}$ | $53.8_{\pm 2.8}$ | $76.6_{\pm 3.1}$ | 70.6 |
| | **SBERT** | $69.9_{\pm 1.8}$ | $72.3_{\pm 0.8}$ | $82.2_{\pm 2.5}$ | $86.3_{\pm 1.2}$ | $69.6_{\pm 0.0}$ | $58.8_{\pm 1.5}$ | $88.5_{\pm 0.9}$ | $59.8_{\pm 2.6}$ | $52.3_{\pm 2.4}$ | $69.1_{\pm 0.8}$ | 70.9 |
| | **SimCSE** | $71.4_{\pm 3.7}$ | $73.7_{\pm 1.6}$ | $82.9_{\pm 0.8}$ | $85.5_{\pm 1.4}$ | $73.2_{\pm 0.0}$ | $59.7_{\pm 1.2}$ | $89.1_{\pm 1.7}$ | $57.8_{\pm 2.2}$ | $51.1_{\pm 2.6}$ | $64.3_{\pm 1.5}$ | 70.9 |
| | **Trans-Encoder** | $70.0_{\pm 1.0}$ | $76.6_{\pm 1.4}$ | $82.1_{\pm 2.0}$ | $84.1_{\pm 1.2}$ | $73.2_{\pm 0.0}$ | $58.0_{\pm 1.0}$ | $89.9_{\pm 1.3}$ | $58.1_{\pm 1.2}$ | $52.0_{\pm 2.4}$ | $70.9_{\pm 2.2}$ | 71.5 |
| | **FADS-ICL** | $90.5_{\pm 1.2}$ | $91.6_{\pm 1.7}$ | $84.2_{\pm 1.9}$ | $89.8_{\pm 0.5}$ | $83.2_{\pm 4.5}$ | $89.6_{\pm 1.7}$ | $98.7_{\pm 0.3}$ | $86.2_{\pm 1.1}$ | $59.2_{\pm 2.0}$ | $91.0_{\pm 0.8}$ | **86.4** |

Table 6: Results for comparison to baselines based on demonstration selection.

| Task | Template | Label Space |
|---|---|---|
| **SST2** | Review: contains no wit , only labored gags<br>Sentiment: negative<br>Review: the film is powerful , accessible and funny .<br>Sentiment: | negative, positive |
| **SUBJ** | Input: the script isn't very good ; not even someone as gifted as hoffman ( the actor ) can make it work .<br>Type: subjective<br>Input: he must do this in secret so that the parents and school personnel know nothing of his plan .<br>Type: | subjective, objective |
| **MPQA** | Review: would not find it at all strange<br>Sentiment: negative<br>Review: as small ( yet acceptable ) as possible<br>Sentiment: | negative, positive |
| **AGNews** | Input: Carlyle Looks Toward Commercial Aerospace (Reuters). "Reuters - Private investment firm Carlyle Group, which has a reputation for making well-timed and occasionally controversial plays in the defense industry, has quietly placed its bets on another part of the market.<br>Type: technology<br>Input: Superstar Kewell remains centre of attention. Socceroo forward Harry Kewell loosens up by tossing around a ball at Bondi beach yesterday. Photo: Craig Golding. There were half a dozen Socceroos standing on a raised platform in Sydney #39;s<br>Type: | world, sports, business, technology |
| **CB** | Premise: It was a complex language. Not written down but handed down. One might say it was peeled down.<br>Hypothesis: the language was peeled down<br>Prediction: False<br>Premise: A: so I don't know if I wasn't drug tested based on that or because the man who hired me didn't request the drug test, because I know that my company does drug testing on occasion. B: Right. Well, for instance, does the company you worked for before have the right or do they have the ability to say, hey, we've already drug tested her and she came up negative. A: Well, no, I don't think they can force another company to not drug test me just by saying that I didn't, I mean,<br>Hypothesis: they can force another company to not drug test her<br>Prediction: | False, True, Neither |
| **CR** | Review: it 's not as stylized as a sony or samsung .<br>Sentiment: negative<br>Review: i went out and got the canon today .<br>Sentiment: | negative, positive |
| **DBPedia** | Input: Geoffrey D. Falksen (born July 31 1982) is an American steampunk writer.<br>Type: artist<br>Input: Monster Night is a 2006 film directed by Leslie Allen and Lorenzo Doumani.<br>Type: | company, school, artist, athlete, politics, transportation, building, nature, village, animal, plant, album, film, book |
| **MR** | Review: "you might say tykwer has done all that heaven allows , if you wanted to make as anti-kieslowski a pun as possible . suffice to say its total promise is left slightly unfulfilled ."<br>Sentiment: negative<br>Review: an alternately raucous and sappy ethnic sitcom . . . you'd be wise to send your regrets .<br>Sentiment: | negative, positive |
| **RTE** | Premise: A man is due in court later charged with the murder 26 years ago of a teenager whose case was the first to be featured on BBC One's Crimewatch. Colette Aram, 16, was walking to her boyfriend's house in Keyworth, Nottinghamshire, on 30 October 1983 when she disappeared. Her body was later found in a field close to her home. Paul Stewart Hutchinson, 50, has been charged with murder and is due before Nottingham magistrates later."<br>Hypothesis: Paul Stewart Hutchinson is accused of having stabbed a girl.<br>Prediction: false<br>Premise: For women earning 22,000 a year, the total pay accumulated after six months maternity leave would be just 5,300 in the UK and 5,850 in Ireland. Entitlements in Germany would also be relatively low, at 5,900, along with those in France, Spain and the Netherlands, all at 6,750. At the other end of the scale, pay received after six months leave in Italy would be 9,150 while in Denmark and Norway it would be as much as 11,000.<br>Hypothesis: Maternity leave varies in Europe.<br>Prediction: | false, true |
| **TREC** | Question: How did serfdom develop in and then leave Russia ?<br>Type: description<br>Question: What is Shakespeare 's nickname ?<br>Type: | description, entity, expression, human, location, number |

Table 7: Templates for ICL. These are minimum cases with only one demonstration example for illustration.