# Task Adaptation Strategies for Language Models

Anmol Agarwal
Georgia Institute of Technology
aagarwal622@gatech.edu

Rayyan Shahid
Georgia Institute of Technology
rshahid9@gatech.edu

## Abstract

*The ABSTRACT is to be in fully-justified italicized text, at the top of the left-hand column, below the author and affiliation information. Use the word "Abstract" as the title, in 12-point Times, boldface type, centered relative to the column, initially capitalized. The abstract is to be in 10-point, single-spaced type. Leave two blank lines after the Abstract, then begin the main text. Look at previous ICCV abstracts to get a feel for style and length. Please note that the title can be up to 512 characters in length. The maximum size of the abstract is 4000 characters.*

[1]

## 1. Introduction/Background/Motivation

In recent years, Large Language Models (LLMs) have shown great performance across various language tasks. However, the models need to be adapted to the specific task in consideration to extract the best possible performance. There are several different adaptation strategies. Two common ones include: *few-shot fine tuning* and *in-context learning (ICL)*. [2] explores these two adaptation strategies in detail and offers a fair comparison between the two. They compare models of different sizes from the OPT and Pythia family for different context lengths on datasets like RTE, MNLI and QQP by measuring both in-domain and out-of-domain accuracies. They conclude that unlike the earlier belief that in-context learning greatly outperforms few-shot fine tuning, if we fairly compare the two strategies, the results are much closer and generally in favor of few-shot fine tuning. Moreover, few-shot fine tuning has an added benefit of reduced runtime latency and correspondingly higher sample processing rate as compared in-context learning because of smaller context sizes at the time of inference. [1] explores a different context-distillation based fine tuning strategy for Language Models. This differs from few-shot finetuning and in-context learning strategies in the fact that apart from using a few high quality labelled samples as the context, it also utilizes a large unlabelled training dataset for fine tuning the model. It has the same benefits in terms of runtime latency as few-shot fine tuning because it doesn't require contexts to be provided at inference time.

In this work, we compare the above task adaptation strategies i.e few-shot fine tuning, in-context learning and context distillation, and also propose two new task adaptation strategies - namely in-context learning with few-shot fine tuning and context distillation with few-shot fine tuning. The motivation behind using in-context learning with few-shot fine tuning is that it allows the model to leverage the context data through both the strategies. The motivation behind using context distillation with few-shot fine tuning is that it should in theory provide the performance of in-context learning with few-shot fine tuning while also providing the runtime latency benefits of context distillation because of the absence of context at inference time.

If this project proves successful, it could significantly enhance the capabilities of existing Large Language Models (LLMs). This would enable them to handle extended dialogue conversations more effectively, answer queries about extensive PDF files, or even auto-complete code while being aware of a large repository. All these improvements could be achieved without the necessity for costly re-training from the ground up or fine-tuning on extensive datasets.

We will describe each of the above strategies in detail in the following section. We will also provide evaluation results for the above strategies for the OPT family on the MNLI dataset for different context lengths. We also report out-of-domain generalization results on the lexical overlap subset of HANS dataset. Please refer to the experimental setup section for more details on the datasets and evaluation metrics used.

## 2. Approach

In this section, we will describe the approach for the different task adaptation strategies explored as part of this work. This includes three existing task adaptation strategies:

1. In-Context Learning (**A1**)

[1] Project GitHub repo

2. Few-Shot Fine Tuning (**A2**)
3. Context Distillation (**A3**)

We also propose two new task adaptation strategies:

1. In-Context Learning with Few-Shot Fine Tuning (**A4**)
2. Context Distillation with Few-Shot Fine Tuning (**A5**)

Apart from the labelled context data, the context distillation based approaches also rely on an unlabelled dataset that would be used for fine tuning the language model based on a distillation loss with respect to a reference model.

### 2.1. In-Context Learning

In-context learning is an approach in which model is given context containing a fixed number of examples for a task with expected output, and then model is given a new example for which the model has to generate an output based on the context.
For example:

```
Question 1: What is the capital of UK?
Answer 1: Paris
Correct: No

Question 2: What is the capital of India?
Answer 2: Berlin
Correct: No

Question 3: What is the capital of Italy?
Answer 3: Rome
Correct: Yes

Question 4: What is the capital of Spain?
Answer 4: Amsterdam
Correct: ?
```

In the above output, the model is given context containing 3 examples and their expected outputs. The model is then asked to predict the output for the 4th example. The model is then evaluated based on the correctness of the output for the 4th example.
One drawback of in-context learning is large context length, as the context length increases, the model becomes slower and has to remember more context.

### 2.2. Few-Shot Fine Tuning

In few-shot fine-tuning, the model is given a few examples for a task with expected output, and then the model is fine-tuned on these examples. The model is then evaluated on a new example for which the model has to generate an output.
This approach may be better than in-context learning from the perspective of model speed and memory usage, as the model only has to process one example rather than entire context.

### 2.3. Context Distillation

Here we have two pretrained models. One model is given context as input, and it has to generate an output for a new example based on the context. We also have another model (initially similar to the first one) which fine tunes on labels generated by first model. In other words, second model is fine tuned on the unlabelled dataset using a distillation loss with respect to the first model.
Let $X$ be unlabeled dataset, $C$ be the context provided to first model to infer the output. This approach is different because few-shot fine-tuning models the distribution $P(X)$ and in-context learning models the distribution $P(X|C)$. We instead want the fine-tuned model to model the distribution $P(X|C)$ despite being given no context. Context distillation helps in achieving this. The first model models the distribution $P(X|C)$ and the second model models the distribution $P(X)$. And our goal is to minimize the KL divergence between the two distributions.
The distillation loss is given by:

$$L_{distillation}(\theta) = KL(P_0(X|C)||P_\theta(X)) \tag{1}$$

Where $P_0(X|C)$ is the distribution modeled by the first model and $P_\theta(X)$ is the distribution modeled by the second model. Through this loss, the second model learns to model the distribution $P(X|C)$ despite being given no context.

### 2.4. In-Context Learning with Few-Shot Fine Tuning

Since it was shown in ¡cite¿ paper that fine-tuning can perform better than in-context learning, we wanted to see if fine-tuned model is given context, will it perform better than in-context learning with pretrained model. In this approach, first model is fine-tuned on few examples and then given context as input to generate output for a new example.
Intuitively speaking, we wanted to validate the hypothesis that since fine-tuned model has already seen examples and being slightly adapted to the task, it might be able to generate better output for the new example given context than the pretrained model.

### 2.5. Context Distillation with Few-Shot Fine Tuning

In this approach, we wanted to see if context distillation can be used in conjunction with few-shot fine-tuning to improve the performance of the model due to the same reasons as in the previous approach. Here, unlike vanilla context distillation, first model used is a fine-tuned one instead of being pretrained. The second model is still pretrained.

| Seed | Train Accuracy | MNLI Eval | HANS Entail | HANS Contradict |
|---|---|---|---|---|
| 0 | 0.48 | 0.528 | 1 | 0 |
| 1 | 0.49 | 0.5204 | 1 | 0 |
| 2 | 0.49 | 0.5248 | 1 | 0 |
| 3 | 0.5 | 0.5289 | 1 | 0 |
| 4 | 0.49 | 0.5225 | 1 | 0 |

Table 1. Accuracy for In-Context Learning (A1) for OPT-125m model

| Seed | Train Accuracy | MNLI Eval | HANS Entail | HANS Contradict |
|---|---|---|---|---|
| 0 | 0.69 | 0.6096 | 0.1846 | 0.791 |
| 1 | 0.63 | 0.6279 | 0.7176 | 0.3126 |
| 2 | 0.59 | 0.5487 | 0.0136 | 0.9876 |
| 3 | 0.69 | 0.6304 | 0.554 | 0.4786 |
| 4 | 0.59 | 0.6087 | 0.5136 | 0.4248 |

Table 2. Accuracy for In-Context Learning (A1) for OPT-1.3b model

| Seed | MNLI Eval | HANS Entail | HANS Contradict |
|---|---|---|---|
| 0 | 0.5766 | 1 | 0 |
| 1 | 0.598 | 0.9526 | 0.0276 |
| 2 | 0.5372 | 1 | 0 |
| 3 | 0.556 | 0.105 | 0.9006 |
| 4 | 0.6462 | 1 | 0 |

Table 3. Accuracy for Few-Shot Fine Tuning (A2) for OPT-125m model

| Seed | MNLI Eval | HANS Entail | HANS Contradict |
|---|---|---|---|
| 0 | 0.6827 | 0.8878 | 0.145 |
| 1 | 0.6573 | 0.9244 | 0.1204 |
| 2 | 0.579 | 1 | 0 |
| 3 | 0.6918 | 0.5916 | 0.4918 |
| 4 | 0.6649 | 1 | 0 |

Table 4. Accuracy for Few-Shot Fine Tuning (A2) for OPT-1.3b model

| Seed | MNLI Eval | HANS Entail | HANS Contradict |
|---|---|---|---|
| 3 | 0.52 | 1 | 0 |

Table 5. Accuracy for Context Distillation (A3) for OPT-125m model

| Seed | MNLI Eval | HANS Entail | HANS Contradict |
|---|---|---|---|
| 3 | 0.6784 | 0.889 | 0.1204 |

Table 6. Accuracy for Context Distillation (A3) for OPT-1.3b model

## 3. Experimental Setup

We compute the performance of the above task adaptation strategies focusing on in-domain and out-of-domain generalization for the natural language inference (NLI) task.

**Model** We choose the OPT model family and specifically the OPT-125m and OPT-1.3b models.

**Datasets** We use the MNLI training dataset for generating the context data. For in-domain generalization, we use the validation set of MNLI dataset. For out-of-domain generalization, we choose the lexical overlap subset of HANS. The MNLI dataset is binarized by removing the neutral samples.

**Evaluation Metrics** We use the following evaluation

| Seed | Train Accuracy | MNLI Eval | HANS Entail | HANS Contradict |
|---|---|---|---|---|
| 0 | 0.49 | 0.5231 | 1 | 0 |
| 1 | 0.51 | 0.5309 | 1 | 0 |
| 2 | 0.49 | 0.5198 | 1 | 0 |
| 3 | 0.51 | 0.4801 | 0 | 1 |
| 4 | 0.57 | 0.5554 | 1 | 0 |

Table 7. Accuracy for In-Context Learning with Few-Shot Fine Tuning (A4) for OPT-125m model

| Seed | Train Accuracy | MNLI Eval | HANS Entail | HANS Contradict |
|---|---|---|---|---|
| 0 | 0.62 | 0.6256 | 0.9386 | 0.0526 |
| 1 | 0.53 | 0.5406 | 0.9986 | 0.0004 |
| 2 | 0.54 | 0.544 | 1 | 0 |
| 3 | 0.66 | 0.6531 | 0.8486 | 0.1484 |
| 4 | 0.66 | 0.6409 | 0.9496 | 0.042 |

Table 8. Accuracy for In-Context Learning with Few-Shot Fine Tuning (A4) for OPT-1.3b model

| Seed | MNLI Eval | HANS Entail | HANS Contradict |
|---|---|---|---|
| 3 | 0.5261 | 1 | 0 |

Table 9. Accuracy for Context Distillation with Few-Shot Fine Tuning (A5) for OPT-125m model

| Seed | MNLI Eval | HANS Entail | HANS Contradict |
|---|---|---|---|
| 3 | 0.6672 | 0.9342 | 0.0794 |

Table 10. Accuracy for Context Distillation with Few-Shot Fine Tuning (A5) for OPT-1.3b model

metrics

- Accuracy on MNLI validation dataset
- Accuracy on HANS-Lexical_overlap-entailment
- Accuracy on HANS-Lexical_overlap-contradiction
- Samples processed per second

**Context Size** We evaluate the training strategies on a context size of 32. For in-context leanrning based strategies, this means providing a context of 32 examples during evaluation. For few-shot fine tuning based strategies, this means providing 32 examples for fine-tuning the model. For context distillation, we randomly sample a subset of 100 examples from the binarized MNLI training dataset and remove their ground-truth labels. The models are then fine tuned using a distillation loss against a reference moodel.

**Seeds** We use 5 different seeds for choosing a random context of 32 examples for each of the following strategies-

1. In-Context Learning
2. Few-Shot Fine Tuning
3. In-Context Learning with Few-Shot Fine Tuning

For Context Distillation, we use the best performing seed from the In-Context Learning strategy. For Context Distillation with Few-Shot Fine Tuning, we use the best performing seed from the In-Context Learning with Few-Shot Fine Tuning strategy.

For the fine tuning based strategies, we fine tune the model for 40 epochs with a batch size of 4.

Our training and evaluation was run on a VM with NVIDIA RTX A6000 48 GB GPU, 2 AMD EPYC 7282 vCPUs and 48 GB RAM.

# 4. Results

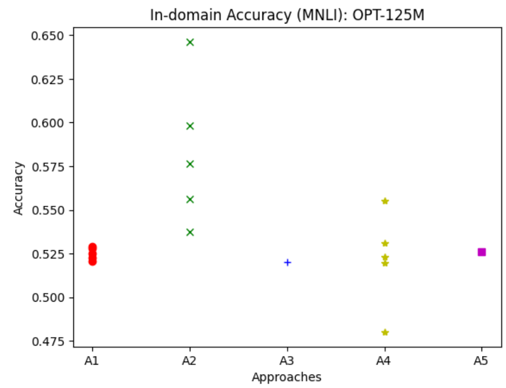


Figure 1. Accuracy on MNLI validation dataset for OPT-125m model

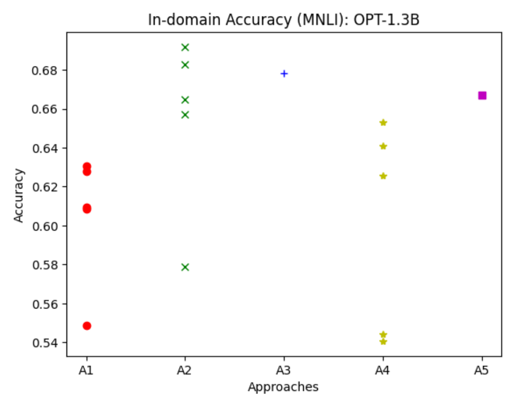


Figure 2. Accuracy on MNLI validation dataset for OPT-1.3b model

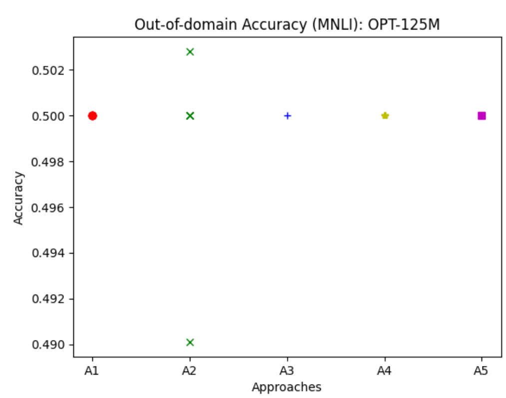


Figure 3. Out of domain accuracy on HANS dataset for OPT-125m

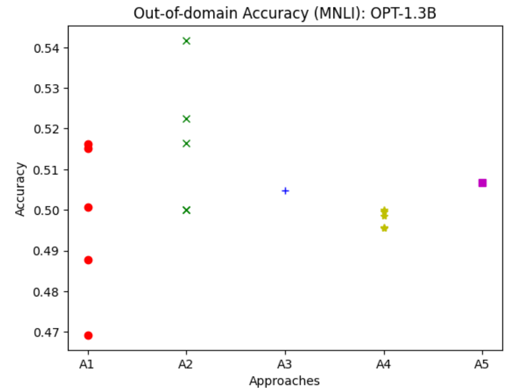


Figure 4. Out of domain accuracy on HANS dataset for OPT-1.3B

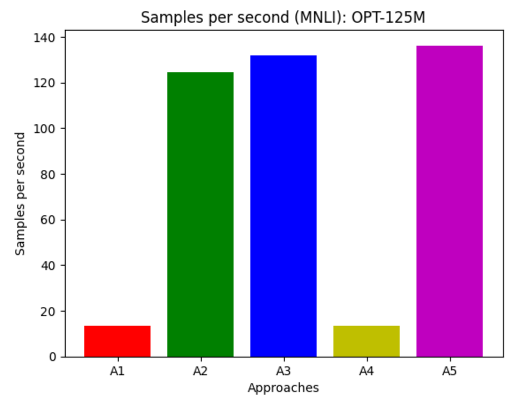


Figure 5. Samples processed per second for OPT-125m model

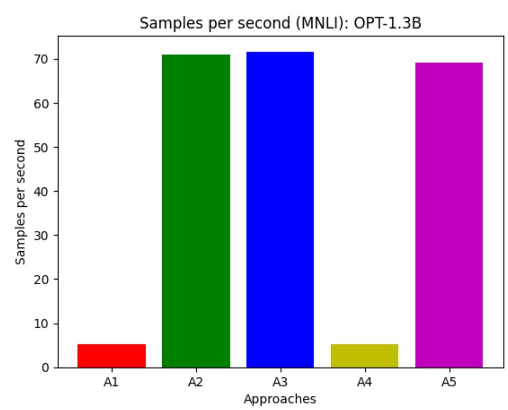


Figure 6. Samples processed per second for OPT-1.3B model

# 5. Challenges

1. **Resource Constraints:** The larger models in the OPT family (for eg. OPT-30b) require a large amount of

GPU memory and training time. Because of our resouce and cost constrainsts, we were unable to evaluate these models. We also had to fix the context size to 32 examples and could only run the experiment for 5 seeds for non context-distillation based approaches. Our experiments on an RTX A6000 48 GB GPU were limited to the OPT-125m and OPT-1.3b models for the NLI task and took a total of $60 spanned over a week.

2. **Environment Setup:** The existing codebase for llmft utilized a docker container for environment setup. This was not possible to setup on PACE VMs. As a result, we had to setup our development environemnt locally and run the experiments on GPUs reserved over tensordock.

3. **Datasets:** Because of time and cost constraints, we only evaluated the strategies for the NLI task on the MNLI and lexical overlap of HANS datasets. Evaluating the models on more datasets would have given us a better understanding of the generalization capabilities of these task adaptation strategies.

# 6. Conclusion

# 7. Work Division

| Task | Anmol | Rayyan |
|---|---|---|
| **A1 experiments** | ✓ | |
| **A2 experiments** | ✓ | |
| **A3 code** | | ✓ |
| **A3 experiments** | | ✓ |
| **A4 code** | ✓ | |
| **A4 experiments** | ✓ | |
| **A5 code** | | ✓ |
| **A5 experiments** | | ✓ |
| **Report** | ✓ | ✓ |

Table 11. Student Contributions

# References

[1] A. Askell, Y. Bai, A. Chen, D. Drain, D. Ganguli, T. Henighan, A. Jones, N. Joseph, B. Mann, N. DasSarma, N. Elhage, Z. Hatfield-Dodds, D. Hernandez, J. Kernion, K. Ndousse, C. Olsson, D. Amodei, T. Brown, J. Clark, S. McCandlish, C. Olah, and J. Kaplan. A general language assistant as a laboratory for alignment, 2021.

[2] M. Mosbach, T. Pimentel, S. Ravfogel, D. Klakow, and Y. Elazar. Few-shot fine-tuning vs. in-context learning: A fair comparison and evaluation, 2023.