Project Report : CS 7643

Yue Yang
Georgia Institute of Technology
yueyang@gatech.edu

Jialin Li
Georgia Institute of Technology
jli3606@gatech.edu

Qi Jiang
Georgia Institute of Technology
qjiang72@gatech.edu

Bo Feng
Georgia Institute of Technology
bfeng66@gatech.edu

Abstract

This paper presents a comprehensive evaluation of efficient fine-tuning methods for Large Language Models (LLMs) in low-resource settings. We compare four distinct approaches: In-Context learning (ICL), vanilla fine-tuning, Low-Rank Adaptation (LoRA), and context distillation, using the OPT model family (125M and 350M parameters) across varying few-shot learning scenarios. Our experiments utilize the RTE dataset for in-domain evaluation and the HANS dataset for out-of-domain generalization assessment. Results demonstrate that while larger models (OPT-350M) generally achieve better performance, they require significantly more computational resources. LoRA proves to be computationally efficient but shows limitations in sequence classification tasks. Context Distillation demonstrates satisfactory performance, though it is not the most optimal. However, the speed at which it computes is commendable, particularly given the accuracy it offers. Our findings provide valuable insights for practitioners seeking to balance model performance with resource constraints in real-world applications. The project GitHub Repo's link: https://github.com/Bo-Feng-1024/CS-7643-Project

1. Introduction/Background/Motivation

In this study, our work is to evaluate and compare the performance of pre-trained large language models using efficient fine-tuning and In-Context Learning approaches. We aim to achieve this by employing: 1) In-Context Learning 2) Vanilla fine-tuning methods, 3) context distillation, and 4) Low-Rank Adaptation of Large Language Models (LoRA) for handling long-form content. We focus on understanding the effectiveness of these methods on tasks involving long dialogues, document comprehension, and code completion across multiple datasets and model architectures.

We explore various efficient fine-tuning approaches and compare them with baseline methods (traditional in-context learning and standard fine-tuning) using the OPT model family. Our experiments maintain consistent training processes and hyperparameters across all comparisons to ensure fair evaluation. We assess performance using both in-domain and out-of-domain metrics, focusing on computational efficiency and model performance.

Current large language models predominantly rely on in-context learning, requiring significant inference-time memory and compute resources. This approach faces practical limitations in real-world applications, particularly when processing long sequences of text or code. Our work is relevant to organizations deploying LLMs for applications like chatbots, document analysis, and code completion. We believe a successful study could significantly reduce computational requirements while maintaining model performance, making advanced LLM capabilities more accessible to organizations with limited resources.

1.1. Datasets

1.1.1 RTE (Recognizing Textual Entailment) Dataset

The RTE dataset was created to evaluate systems' ability to recognize textual entailment - whether one text fragment logically follows from another. It was developed through the PASCAL RTE Challenges (2005-2011) to address the fundamental need in natural language processing to understand semantic relationships between text pairs. Each RTE dataset contains pairs of text snippets with binary annotations indicating whether the second text (hypothesis) can be inferred from the first (premise) [1].

1.1.2 HANS (Heuristic Analysis for NLI Systems) Dataset

The HANS dataset,[2] created by McCoy, Min, and Linzen at Johns Hopkins University, contains 30,000 samples designed to test NLI (Natural Language Inference) systems' reasoning capabilities. Its primary purpose was to expose and evaluate models' reliance on superficial heuristics in NLI tasks. The dataset systematically challenges three common heuristics: lexical overlap, subsequence, and constituent structure. This dataset was developed with support from the National Science Foundation Graduate Research Fellowship Program and has become crucial for evaluating the robustness of NLI models against spurious correlations.

2. Approach

2.1. In-Context Learning

In-context learning (ICL) is a task adaptation strategy. Unlike other methods, ICL does not update the weights of the pretrained model [3]. Instead, it adapts a model to a task by conditioning it on a sequence of demonstrations. A demonstration is simply an input x accompanied by its ground-truth label y, both of which are converted to a specific format using a pattern and a verbalizer.

2.2. Few-Shot Learning

Few-shot learning is a training paradigm designed to enable models to generalize effectively to new tasks with very limited labeled data. In this work, we incorporate the few-shot learning paradigm by training the model with N-shot samples, where N represents the number of labeled examples per class available for training. A small labeled dataset (e.g., N=2,32,128 examples per class) is used to fine-tune the pre-trained model. These samples represent the data available for training in a few-shot setting. To further assess robustness, we repeat the training and evaluation process for each N-shot setup multiple times (K= 3 or 10) , as this helps mitigate the variability caused by the small training datasets. By default, we use the GPT-3 prompt patten to construct these shots.

2.3. Fine-Tuning Methods

2.3.1 Vanilla Fine-Tuning

Vanilla fine-tuning is a straightforward approach to adapting pre-trained language models for more specific tasks by using backpropagation and gradient descent. This process starts with a pre-trained model that has already learned general language understanding from a large text corpus, then adds a task-specific layer (such as a classification head), and continues training the entire model end-to-end on the target dataset.

While Vanilla Fine-Tuning allows full model adaptation to the target task, it comes with significant computational costs and memory requirements since all parameters are being updated. Additionally, vanilla fine-tuning risks catastrophic forgetting of pre-trained knowledge and may overfit on small datasets [3].

2.4. Parameter-Efficient Modeling using Low-Rank Adaptation (LoRA)

PEFT methods aim to minimize the number of parameters updated during fine-tuning. This makes PEFT computationally lightweight and scalable. Among the various PEFT methods, LoRA (Low-Rank Adaptation) stands out for its ability to efficiently adapt large pre-trained models to new tasks.[4]

LoRA introduces trainable low-rank matrices that are added to the frozen weight matrices of the model [5] This is formulated as follows: $W' = W + A \cdot B$, where W is the original weight matrix, B and A are the low-rank matrices, and W' is the modified weight matrix. These matrices capture task-specific information while keeping the majority of the pre-trained model's parameters intact, ensuring that the model's original knowledge is preserved.

2.5. Context Distillation

Context distillation is a method of fine-tuning a model to replicate the behavior of a prompted model, eliminating the need for prompting during inference. This approach solves problems associated with limited context window sizes and the computational inefficiency of prompting [6].

Context distillation is a process comprising three principal stages. Initially, training data is generated by prompting a large language model with the desired context (e.g., the helpful, honest, and harmless (HHH) prompt). This involves storing token indices and top predicted probabilities to capture the prompt's behavior. Subsequently, a distinct model is adapted to minimize the Kullback–Leibler divergence between its predictions and the initial model's probability distribution, thereby acquiring the same behavioral patterns. Ultimately, the fine-tuned model internalizes the prompt, enabling it to generate responses aligned with the desired behavior without requiring the prompt during inference.

We refer to the code from Srinivasan et al. (2024) [7] and modify it to experiment with RTE as the in-domain dataset and HANS as the out-of-domain dataset. Due to time constraints, we're only experimenting with the opt 125M model. The number of shots is 2, 32, and 128.

3. Experiments and Results

3.1. Data Pre-processing

Each pair of sentences from the RTE and Hans dataset is typically converted into a prompt-based question to leverage the model's pre-trained capabilities on language inference. Both RTE and the HANS dataset sentences are tokenized and then are padded or truncated to ensure uniform input sizes.

The model outputs logits or probabilities for each class. These are raw scores that indicate the likelihood of each class being the correct answer.

3.2. In-Context Learning

3.2.1 Problem Structure and Model Design

This experiment assesses the in-context learning capabilities of the OPT-125M language model on natural language inference tasks with varying numbers of demonstration examples, referred to as ”shots.” The evaluation is conducted on both in-domain (RTE) and out-of-domain (HANS) performance. The number of demonstration examples (shots) varies: 2, 32, and 128. The experiment employs a consistent GPT-3 pattern prompt template. To implement the ICL experiment, we utilize balanced sampling to ensure equal representation of each label class and shuffle demonstrations to enhance robustness.

3.2.2 Measurement of Success

As table 1 shown, the results demonstrate precise and uniform consistency across all shot counts. RTE accuracy was found to be 47.29% for 2, 32, and 128 shots, while HANS accuracy was observed to be 50.00% for the same shot counts.

The identical results are explained as follows. With regard to context processing, the identical scores across different shot counts suggest that the model is not effectively utilising the demonstration examples. Additionally, it is possible that the context is being truncated for larger shot counts. From the perspective of the model itself, OPT-125M is a relatively small language model (125M parameters), which may not have sufficient capacity to benefit from an increase in the number of demonstrations.

Num of Shots	In-Domain Accuracy	Out-of-Domain Accuracy
2	0.4729	0.5000
32	0.4729	0.5000
128	0.4729	0.5000

Table 1. ICL in-domain and out-of-domain accuracy

3.3. Vanilla FT

In this section, we outline the structure of the problem, the design of our experiments as well as results using vanilla fine-tuning method.

3.3.1 Problem Structure and Model Design

Vanilla fine-tuning serves as a straightforward benchmark method to allow us to cross-comparison with other efficient fine-tuning approaches, such as context distillation, LoRA, and in-context learning. The result in this section provides a baseline for evaluating the effectiveness of alternative strategies.

We begin with a pre-trained large language model (Facebook OPT-125M), trained on a broad and diverse dataset.

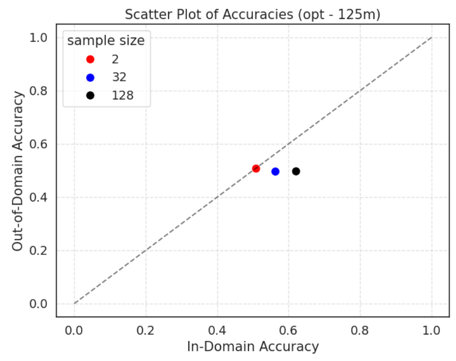

Figure 1. Comparison of In-Domain accuracy on RTE vs Out-of-Domain accuracy on HANS (OPT-125M)

The fine-tuning process involves adapting this general-purpose model specifically to the task domain, starting with the Recognizing Textual Entailment (RTE) dataset as described earlier. Subsequently, we fine-tune the model weights using the same architecture and training objective as the base model to evaluate its performance on an out-of-domain task, namely the HANS dataset. Model Hyperparameters are listed as follows:

- **Loss function:** the Cross-Entropy Loss, as it is the most commonly used loss function for sequence classification tasks.
- **Optimizer:** We used the AdamW optimizer which is common choice of optimizer to perform fine tune tasks.
- **Batch size:** 32.
- **Learning rate:** 1e-5.
- **Few shot sample size:** 2, 32, 128
- **Number of epoch:** 40

3.3.2 Experiments

In-Domain vs. Out-of-Domain Accuracy Across Few-Shot Sample Sizes Figure 1 plots the in-domain and out-of-domain accuracy of the vanilla fine-tuned Facebook OPT-125M model under varying sample sizes ($n = 2, 32, 128$). The results show that the in-domain accuracy on the RTE dataset improves significantly, increasing from 50.3% to 59.9% as the sample size of few shot grows. However, the out-of-domain accuracy on the HANS dataset exhibits minimal improvement with the OPT-125M model (consistently around 49.9%). We suspect that the OPT-125M model is too small to fully demonstrate the benefits of vanilla fine-tuning. This limitation is explored further in the next experiment.

Impact of Model Size To assess the effect of model size, we conducted the same experiment using both the OPT-125M and the larger OPT-350M models. Table 2 and Table 3 respectively summarize the average accuracy and model loss for both models, which demonstrates that the OPT-350M model consistently outperforms the OPT-125M model on in domain RTE dataset when few shots sample size is small. However, the result is mixed when few shot sample size is large and on out-of-domain HANS data.

Additionally, we evaluated run-time performance: Table 4 shows that the larger OPT-350M model requires significantly more computational time (2.3x longer). The total runtime for the OPT-125M model was approxiamtely 2.6 hours, compared to 8.7 hours for the OPT-350M model.

Model Size	Few Shots	In-Domain Accuracy	Out-of-Domain Accuracy
125M	2	0.5032	0.4994
125M	32	0.5271	0.4988
125M	128	0.5993	0.5000
350M	2	0.5300	0.4868
350M	32	0.5451	0.4870
350M	128	0.5884	0.5001

Table 2. Comparison of Model Accuracy for different models and few-shot settings.

Model Size	Few Shots	In-Domain Loss	Out-of-Domain Loss
125M	2	1.4172	0.7631
125M	32	2.3221	0.7665
125M	128	3.2664	1.0285
350M	2	3.1801	0.9275
350M	32	2.2612	1.1395
350M	128	3.1948	2.0195

Table 3. Comparison of Model Loss for different models and few-shot settings.

Model Size	Few Shots	In-Domain Time (s)	Out-of-Domain Time (s)
125M	2	1.964	205.909
125M	32	1.966	206.288
125M	128	1.962	206.375
350M	2	6.131	691.156
350M	32	6.595	690.782
350M	128	6.535	692.431

Table 4. Comparison of Run Time for in-domain and out-of-domain settings across different models and few-shot settings.

Hyperparameter Tuning Conceptually, fine-tuning a pre-trained model requires a smaller learning rate than pre-training, as the base model already contains useful knowledge and only needs incremental adjustments.

To test this hypothesis, we experimented with a larger learning rate of 1e-4 compared to the default value of 1e-5. Figure 2 reveals a degradation in both in-domain and out-of-domain performance with the larger learning rate, supporting the preference for smaller learning rates during fine-tuning. These findings emphasize the importance of carefully tuning hyperparameters to achieve optimal performance when fine-tuning pre-trained large language models.

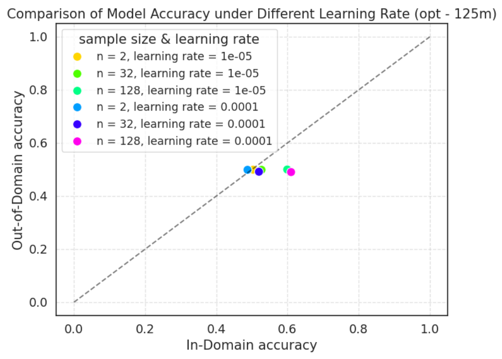

Figure 2. Comparison of Model Accuracy under different learning rate and model of different sizes (OPT-125M and 350M)

3.4. Effect of using parameter-efficient modeling

3.4.1 Problem Structure and Model Design

Problem Structure: The task required fine-tuning the OPT-125M/350M model for textual entailment (RTE) and adversarial generalization (HANS) with few-shot learning. The challenge lay in balancing efficiency and generalization across domains.

Model Design: By keeping the pre-trained model weights frozen and adding trainable low-rank matrices, LoRA provided a computationally efficient means of adapting the model to new tasks. This structure reflects the problem's requirements for efficiency and adaptability.

Learned Parameters: The LoRA matrices were the only trainable parameters, while the rest of the model remained frozen. This separation allowed for efficient fine-tuning without altering the large pre-trained model.

- The LoRa rank is chosen based on balanced performance using in-domain and out-of-domain accuracy as evaluation (few-shot sample number = 2) (Table 5). Although rank 128 achieved the best result, we settled on 4 to reduce computational cost.

LoRA Rank (r)	In-Domain Accuracy	Out-of-Domain Accuracy
1	0.5451	0.5182
2	0.5487	0.5184
4	0.5487	0.5189
8	0.5415	0.5197

Table 5. Comparison of In-Domain and Out-of-Domain Accuracy for Different LoRA Ranks (few-shot=2).

- The LoRA α value acts as a scaling factor for the low-rank updates during training. A moderate value 8, ensures the updates are not too large, preserving the stability of the pre-trained model weights.

- The LoRA dropout parameter (0.1) ntroduces regularization by randomly setting some elements of the low-rank updates to zero during training.

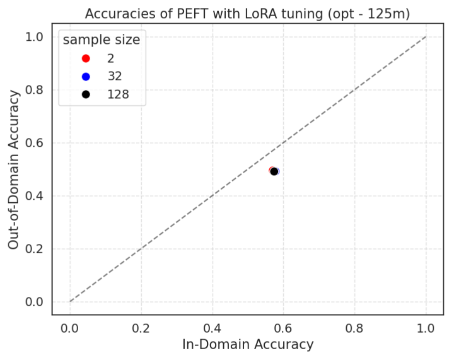

Figure 3. In-domain (RTE) and out-of-domain (HANS) performance for peft + LoRA with OPT-125M

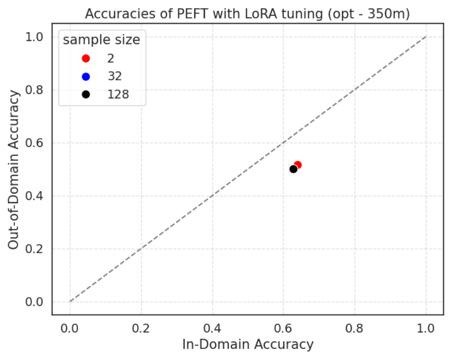

Figure 4. In-domain (RTE) and out-of-domain (HANS) performance for peft + LoRA with OPT-350M

Loss function and optimizer We used the Cross-Entropy Loss, as it is the most commonly used loss function for sequence classification tasks. The AdamW optimizer was used to update the LoRA parameters, providing robust convergence properties for transformer-based architectures.

3.4.2 Measurement of Success

Performance: In Figure 3 and Figure 4, we observed that the in-domain accuracy ranges from approximately 0.475 and 0.650, while the out-of-domain accuracy is generally lower, ranging between 0.475 and 0.625, indicating the model performs better on in-domain (RTE) tasks compared to the out-of-domain (HANS) tasks. The proximity of points below the diagonal line suggests the models perform better on the in-domain (RTE) dataset compared to the out-of-domain (HANS) dataset.

Model Size: The larger OPT-350M model consistently achieves higher accuracies for both in-domain and out-of-domain tasks compared to the smaller OPT-125M model.

Few-shot: For both models (OPT-125M and OPT-350M), different few-shot settings (n = 2, 32, 128) show clustering of results, with only slight variations in both in-domain and out-of-domain accuracies. This suggests that the few-shot size has a limited impact on model performance within the tested range.

Other assessments: Loss and running time are summarized in Table 6 and Table 7. The total running time is 4 hr and 49 sec for OPT-125M and 13 hrs and 17 min for OPT-350M.

n	Iter	In-Domain Loss	Out-of-Domain Loss	Running Time (s)
2	1	0.69365	0.71326	1708.39
2	2	0.68994	0.70583	1679.97
2	3	0.68994	0.70583	1676.13
32	1	0.68987	0.70560	1656.11
32	2	0.68987	0.70560	1655.48
32	3	0.68987	0.70560	1655.33
128	1	0.68960	0.70545	1652.46
128	2	0.68960	0.70545	1652.94
128	3	0.68960	0.70545	1656.33

Table 6. Experiment Data With Running Times and Losses (OPT-125M)

n	Iter	In-Domain Loss	Out-of-Domain Loss	Running Time (s)
2	1	0.6399	0.6881	5342.43
2	2	0.7446	0.8851	5331.42
2	3	0.7446	0.8851	5316.36
32	1	0.74507	0.88738	5295.59
32	2	0.74507	0.88738	5285.68
32	3	0.74507	0.88738	5299.82
128	1	0.74403	0.88767	5331.48
128	2	0.74403	0.88767	5320.18
128	3	0.74403	0.88767	5329.16

Table 7. Experiment Data With Running Times And Losses (OPT-350M)

3.5. Effect of using context distillation

3.5.1 Problem Structure and Model Design

This study investigates the effectiveness of context distillation using the OPT-125M model for natural language inference tasks, with a particular emphasis on evaluating its performance on the RTE (Recognizing Textual Entailment) dataset and its capacity for generalizing across different datasets. Both the teacher and student models are configured with dropout probabilities of 0.1 for both hidden layers and attention, and half-precision (FP16) training has been implemented for memory efficiency. These models are based on the OPT-125M model.

Training Setup The dataset sizes were 2, 32, and 128 examples per class. To ensure statistical reliability, 10 runs were conducted for each dataset size. Training was performed for 40 epochs, with a batch size of 8 and gradient accumulation steps of 4. The learning rate was set to 1e-5, with a warmup period of 10% of the total steps.

Loss Function The loss function is an equally weighted average of the distillation loss and the classification loss. The distillation loss employs the KL divergence between the outputs of the teacher and the student, while the classification loss utilizes the standard cross-entropy.

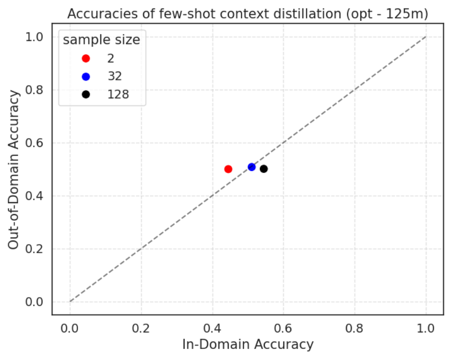

Figure 5. In-domain (RTE) and out-of-domain (HANS) performance for Context Distillation with OPT-125M

Optimization The AdamW optimizer was utilized with a weight decay value of zero, FP16 training enabled, gradient accumulation implemented for an effective batch size of 32, and device mapping was employed to facilitate optimal resource utilization.

3.5.2 Measurement of Success

As the figure 5 shown, The out-of-domain accuracy of all sample sizes is approximately 0.50, exhibiting minimal variation. As the sample size increases, there is a slight improvement in in-domain accuracy, with the points becoming more concentrated. This indicates that the model performs better in-domain (RTE) than out-of-domain (HANS), as all points fall below the diagonal line.

Loss and average running time are summarised in table in the table 8. The total runtime for OPT-125M is 4 hours, 27 minutes and 59 seconds.

n	In-Domain Accuracy	Out-of-Domain Accuracy	Execution Time (s)
2	0.4451	0.5000	424.6
32	0.5108	0.5079	487.3
128	0.5448	0.5011	696.0

Table 8. In-Domain and Out-of-Domain Accuracy and Execution time for Context Distillation

4. Related work

Recent studies have extensively explored various approaches to fine-tuning large language models. Hu et al.[5] introduced LoRA as an efficient adaptation method, demonstrating competitive performance while significantly reducing parameter count through low-rank decomposition. Our experiments with LoRA confirm these efficiency benefits, though we observed some limitations in sequence classification tasks with OPT models.

The context distillation approach builds upon work by Mosbach et al.[8], who conducted fair comparisons between few-shot fine-tuning and in-context learning. While their study focused primarily on in-domain performance, our work extends this analysis to out-of-domain scenarios, revealing that context distillation can outperform traditional fine-tuning methods in certain setups while requiring substantially less computational resources.

McCoy et al.[2]investigated syntactic heuristics in natural language inference, particularly through the HANS dataset. Their work highlighted the importance of evaluating model performance on out-of-domain data, which influenced our experimental design. Our results extend their findings by showing that different fine-tuning methods with both out-of-domain and in-of-domain evaluations.

Our work differs from previous studies by providing a comprehensive comparison of multiple fine-tuning approaches specifically in low-resource settings, evaluating both in-domain and out-of-domain performance.

5. Conclusion

Our comprehensive analysis of efficient fine-tuning methods reveals several key findings for deploying LLMs in resource-constrained environments. First, we demonstrate that model size significantly impacts both performance and computational requirements, with OPT-350M showing better accuracy but requiring 2.7x longer training time compared to OPT-125M. Second, our experiments with LoRA confirm its computational efficiency but highlight limitations in sequence classification tasks, particularly in out-of-domain generalization.

The study also improves in-domain accuracy for smaller models but shows diminishing returns for larger models. This finding has important implications for resource allocation in practical applications. Furthermore, we show that our implementation of context distillation maintains reasonable in-domain accuracy that improves with increasing sample size (from 44.51% with 2 shots to 54.48% with 128 shots) while achieving fair performance, and its computational speed makes it an attractive option for resource-constrained environments, with a total runtime of only 4 hours and 28 minutes.

6. Work Division

The delegation of tasks among team members is summarized in the table below. Each team member contributed to specific parts of the project to ensure efficient progress and quality of work. 9

References

[1] I. Dagan, O. Glickman, and B. Magnini, "The pascal recognising textual entailment challenge," in *Machine learning challenges workshop*, pp. 177–190, Springer, 2005. 1

Student Name	Contributed Aspects	Details
Yue Yang	Implementation, Analysis, and Report Writing	Implemented peft+LoRA mechanism to improve fine-tuning. Tried both 125M and 350M. Analyzed these results, made figures, and wrote the relevant section in the report.
Qi Jiang	Implementation, Analysis, and Report Writing	Explored vanilla FT baseline using various model sizes and hyperparameter tuning.
Jialin Li	Literatue Review, Implementation and Report Writing	Explored vanilla FT baseline using RTE; paper written and formate.
Bo Feng	Implementation, Analysis, Literatue Review, Organizing Meetings, and Report Writing	Implemented Context Distillation and vanilla FT. Replicated ICL experiment results. Analyzed results, made tables and figures on experiment results, and wrote the summarization.

Table 9. Contributions of team members.

[2] R. McCoy, "Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference," *arXiv preprint arXiv:1902.01007*, 2019. 1, 6

[3] M. Mosbach, M. Andriushchenko, and D. Klakow, "On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines," *arXiv preprint arXiv:2006.04884*, 2020. 2

[4] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *International conference on machine learning*, pp. 2790–2799, PMLR, 2019. 2

[5] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021. 2, 6

[6] A. Askell, Y. Bai, A. Chen, D. Drain, D. Ganguli, T. Henighan, A. Jones, *et al.*, "A general language assistant as a laboratory for alignment," *arXiv*, vol. 2112.00861v3, 2021. 2

[7] K. P. V. Srinivasan, P. Gumpena, M. Yattapu, and V. H. Brahmbhatt, "Comparative analysis of different efficient fine tuning methods of large language models (llms) in low-resource setting," *arXiv preprint arXiv:2405.13181*, 2024. 2

[8] M. Mosbach, T. Pimentel, S. Ravfogel, D. Klakow, and Y. Elazar, "Few-shot fine-tuning vs. in-context learning: A fair comparison and evaluation," *arXiv*, vol. 2305.16938v2, 2023. 6