# Few-shot Fine-tuning vs. In-context Learning: A Fair Comparison and Evaluation

**Marius Mosbach**[1] **Tiago Pimentel**[2] **Shauli Ravfogel**[3] **Dietrich Klakow**[1] **Yanai Elazar**[4,5]
[1]Saarland University, Saarland Informatics Campus, [2]University of Cambridge,
[3]Bar-Ilan University, [4]Allen Institute for Artificial Intelligence, [5]University of Washington
`mmosbach@lsv.uni-saarland.de`

## Abstract

Few-shot fine-tuning and in-context learning are two alternative strategies for task adaptation of pre-trained language models. Recently, in-context learning has gained popularity over fine-tuning due to its simplicity and improved out-of-domain generalization, and because extensive evidence shows that fine-tuned models pick up on spurious correlations. Unfortunately, previous comparisons of the two approaches were done using models of different sizes. This raises the question of whether the observed weaker out-of-domain generalization of fine-tuned models is an inherent property of fine-tuning or a limitation of the experimental setup. In this paper, we compare the generalization of few-shot fine-tuning and in-context learning to challenge datasets, while controlling for the models used, the number of examples, and the number of parameters, ranging from 125M to 30B. Our results show that fine-tuned language models *can* in fact generalize well out-of-domain. We find that both approaches generalize similarly; they exhibit large variation and depend on properties such as model size and the number of examples, highlighting that robust task adaptation remains a challenge. [1]

## 1 Introduction

Adapting a pre-trained language model to a target task is of high practical importance to the natural language processing (NLP) community (as seen in Peters et al., 2018; Howard and Ruder, 2018; Devlin et al., 2019; Brown et al., 2020, *inter alia*). Among the commonly used *task adaptation* strategies, two stand out: *fine-tuning* (FT) and *in-context learning* (ICL).[2]

[1]Code available at: https://github.com/uds-lsv/llmft.

[2]We describe both these strategies in further detail in Section §2. In short, fine-tuning a model involves a supervised learning setup on a target dataset; while ICL involves prompting a model with a series of input–label pairs, without updating the model's parameters.

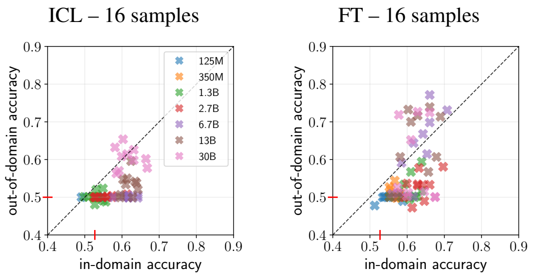


Figure 1: In-domain (RTE) and out-of-domain performance (HANS) for in-context learning (ICL) and fine-tuning (FT) with OPT models of various sizes. We fine-tune models using pattern-based fine-tuning. We report results using 10 different data seeds. When using 16 samples, ICL's performance with a 30B model is comparable to that of FT with smaller models (6.7B) and for most model sizes, FT outperforms ICL (see Table 1a for significance tests). — in the x- and y-axes indicates majority class accuracy.

Both approaches come with pros and cons: ICL reuses a single pre-trained model for various downstream tasks, allows specifying the desired behavior via natural language, and has recently shown impressive results on challenging reasoning tasks (Brown et al., 2020; Wei et al., 2022; Press et al., 2022b). However, the model's context size limits the number of demonstrations that can be used. For instance, using 32 randomly selected examples from the RTE dataset (Dagan et al., 2006) already exceeds the context size of OPT models (Zhang et al., 2022).[3] In addition, ICL is highly sensitive to the format and order of its inputs (Lu et al., 2022; Min et al., 2022). FT, on the other hand, typically results in a single specialized model per task,[4] and can be applied to training sets of arbitrary size. However, such models are sensitive to initialization

[3]While GPT-3 and OPT both have a context size of 2048 tokens, more recent models such as GPT-4 (OpenAI, 2023) – which has been developed concurrently to this work – support larger contexts of up to 8192 tokens.

[4]Parameter-efficient FT methods (e.g. Ben Zaken et al. (2022); Hu et al. (2022)) address this issue and allow to re-use most of the pre-trained weights across tasks.

(Dodge et al., 2020) and can suffer from instability during training (Mosbach et al., 2021).

For text classification tasks, where both strategies often lead to similar performance on in-domain data (when using the same amount of data), recent works have argued that ICL leads to better out-of-domain (OOD) generalization (Si et al., 2023; Awadalla et al., 2022). However, these comparisons of generalization abilities were not conducted under equal conditions. Most studies compare the ICL abilities of large models (e.g. GPT-3, 175B; Brown et al., 2020) to the FT abilities of much smaller models (e.g. RoBERTa-large, 350M; Liu et al., 2019). These comparisons raise the question of whether FT indeed leads to weaker OOD generalization than ICL, or whether this is just a byproduct of the experimental setup. In Figure 1, we show this is indeed the case: when given only 16 examples, fine-tuning a 6.7B parameters model already achieves similar results to ICL with a 30B model, and FT performance keeps improving with larger models.[5] Moreover, we show in Section 4.1 that fine-tuning performance improves even further when training on more data.

In this paper, we compare ICL and FT on an **equal footing** (§3). We compare both strategies using the same model (OPT; Zhang et al., 2022), the same number of parameters (from 125M to 30B), and the same number of examples. Our results and analyses (§4) show that both approaches often achieve comparable results. Both methods are unstable and can perform badly on in-domain and OOD data due to training instability, or prompt choice. We also find that both approaches improve as we increase model size, and that, for the models and datasets we consider, FT often generalizes even better than ICL. Notably, this is in contrast to prior work (§7), highlighting the need for fair comparisons of task adaptation strategies. Based on our findings, we discuss the strengths and limitations of FT and ICL (§6), which can inform when to use and how to get the most out of each method.

## 2 Background

### 2.1 Fine-tuning

*Pattern-based fine-tuning* (PBFT) is a recently proposed FT approach that uses the pre-trained language modeling head[6] instead of a randomly initialized classifier (as used in standard fine-tuning; Howard and Ruder 2018; Devlin et al. 2019), to obtain predictions (Schick and Schütze, 2021; Gao et al., 2021b, *inter alia*). Compared to vanilla FT, we have to specify an *input pattern* (to cast the task as a language modeling problem) and define a *verbalizer* (which maps tokens in the pre-trained model's vocabulary to labels; Schick et al., 2020). For example, a NLI pattern might look as follows: `{premise} Question: {hypothesis} Yes or No?`, and the verbalizer will use `Yes` and `No` as tokens. Given these inputs and targets, model parameters are fine-tuned as usual. This method has been shown to be efficient for few-shot learning despite having no advantage over vanilla FT when the number of examples is large (Tam et al., 2021; Logan IV et al., 2022).

### 2.2 In-context learning

*In-context learning* (ICL) is a task adaptation strategy that does not update the weights of the pre-trained model (Brown et al., 2020); instead, ICL adapts a model to a task by conditioning it on a sequence of *demonstrations*. A demonstration typically refers to an input $x$ accompanied by its ground-truth label $y$, both of which have been converted to a specific format using a *pattern* and a *verbalizer* (similar to PBFT). ICL thus feeds the model a sequence of such demonstrations, followed by the test input (modified by applying the pattern transformation). The language model is then expected to predict the label of this final data point.[7] Recent work has argued that ICL leads to better out-of-domain performance, when compared to FT (Si et al., 2023; Awadalla et al., 2022). We show that this often does not hold.

## 3 A fair comparison of FT and ICL

We perform a fair comparison of task adaptation via FT and ICL, focusing on in-domain and OOD generalization. We compare them in the few-shot setting using the same models. In the following paragraphs, we provide details about our setup.

**In-domain generalization** We measure in-domain generalization by measuring accuracy on the validation set of each dataset. This is a common practice in analysis works, and used in previous work (Utama et al., 2021; Bandel et al., 2022).

[5]Table 1a presents significance tests for these results.

[6]In the case of encoder-only masked language models, such as BERT, this is usually an MLP layer. In the case of decoder-only models, such as OPT, this is a linear projection.

[7]The evaluation only considers the probabilities assigned to the verbalizer tokens, ignoring any probability mass assigned to other tokens. See §3 for details.

**Out-of-domain generalization** We consider OOD generalization under *covariate shift* (Hupkes et al., 2022). Specifically, we focus on generalization to *challenge datasets*, designed to test whether models adopt a particular heuristic, or make predictions based on spurious correlations during inference (McCoy et al., 2019; Elazar et al., 2021).

**Models** We run all our experiments using 7 different OPT models (Zhang et al., 2022) ranging from 125 million to 30 billion parameters, all of which have been trained on the same data. This allows us to study the effect of model size on performance without the confound of using different training data.[8]

**Tasks and datasets** We focus on two classification tasks in English: natural language inference (NLI) and paraphrase identification. For NLI, we use MNLI (Williams et al., 2018) and RTE (Dagan et al., 2006) as in-domain datasets, and evaluate OOD generalization on the lexical overlap subset of HANS (McCoy et al., 2019).[9] We binarize MNLI by removing the neutral examples[10] which allows us to better compare MNLI with RTE (which only has two labels). For paraphrase identification, we train on QQP (Sharma et al., 2019) and evaluate OOD generalization on PAWS-QQP (Zhang et al., 2019). Given the large size of the QQP validation set (more than 300k examples), we randomly select 1000 validation examples.

**Few-shot setup** We follow the same procedure for both approaches. We randomly sample $n \in \{2, 16, 32, 64, 128\}$ examples from the in-domain training set of a given dataset (unless stated otherwise).[11] Due to the high sensitivity of both approaches to the used pattern, as well as to the ordering of the demonstrations in ICL (Webson and Pavlick, 2022; Lu et al., 2022), we sample 10 different sets of examples for each $n$. We also experiment with 3 different patterns, resulting in 30 runs per $n$ and adaption method.[12] Table 5 in Appendix A.3 provides an overview of the patterns and verbalizers for each task.

[8]OPT 30B is the largest model we were able to fit given our resources.

[9]Due to similar trends on different HANS subsets in preliminary experiments, we focus on the lexical overlap subset.

[10]We compare this to merging the neutral and contradiction classes in Appendix B.3, and obtain very similar results.

[11]We sample an equal number of examples per label.

[12]Except for QQP, where we experiment with only 2 patterns, as one of the patterns is not applicable.

**FT setup** We perform few-shot PBFT using a minimal pattern (Logan IV et al., 2022), which simply adds a question mark at the end of every example. For the NLI verbalizer, we use `Yes` and `No`, which we map to the task's labels `entailment` and `not-entailment` respectively. For QQP, we also use `Yes` and `No` and map them to `not-duplicate` and `duplicate`.[13] We follow the recommendations of Mosbach et al. (2021) and fine-tune all models for 40 epochs using a learning rate of $10^{-5}$ which increases linearly (warmup) for the first $10\%$ of the training steps and is kept constant afterward. Details of all hyperparameters are provided in Appendix A.5.

**ICL setup** Given OPT's fixed context size of 2048 tokens we are limited in the number of examples used for demonstration. Our main experiments focus on 16 demonstrations, but we also present additional experiments using 2 and 32 demonstrations in Appendix B.[14]. We consider a prediction to be correct if the probability assigned to the verbalizer token of the ground-truth label is larger than the probability of the other verbalizer token. We use the same verbalizer tokens as for fine-tuning.

## 4 Results

We present the results for in-domain and OOD model performance in Figure 2, comparing both ICL and FT. We perform task adaptation using 16 examples for both strategies. For ICL, we provide additional results that demonstrate the importance of choosing the right pattern and number of demonstrations in Appendix B.2. For FT, we provide more details, ablations and discussion of various choices later in this section.

**In-domain performance** For MNLI and RTE, both ICL and FT exhibit in-domain performance above the majority baseline for most model sizes. Focusing on ICL, MNLI and RTE in-domain performance improves as model size increases. On MNLI the largest model (30B) obtains an average performance of $71.4\%$ and a maximum performance of $74.9\%$. On RTE, ICL with the same model achieves an average and maximum performance of $61.7\%$ and $66.8\%$ respectively. On QQP, the trend of improved performance with increasing model size

[13]Preliminary experiments showed that `Yes` and `No` is a strong verbalizer for binary classification tasks. This is consistent with previous findings (Webson and Pavlick, 2022).

[14]With the exception of RTE, where 32 examples do not fit OPT's context size

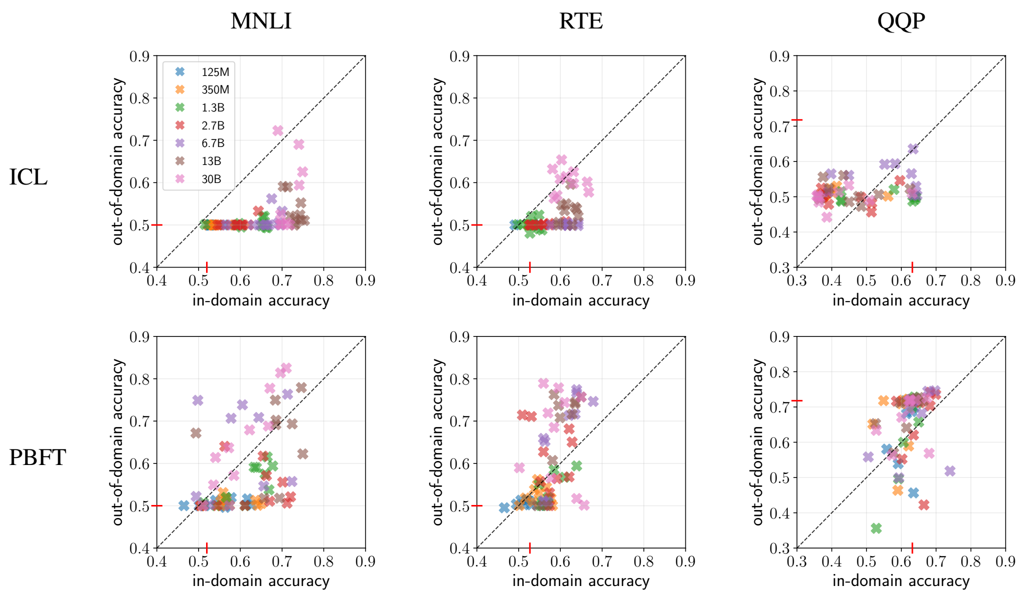


Figure 2: ICL and FT results for OPT models of various sizes. For each approach, we use 16 examples and perform model selection according to OOD performance. We plot 10 runs per model size which differ only in the data seed. — in the x- and y-axis indicates majority class accuracy.

| | | **FT** | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | **125M** | **350M** | **1.3B** | **2.7B** | **6.7B** | **13B** | **30B** |
| **ICL** | **125M** | −0.00 | 0.01 | 0.02 | 0.03 | 0.12 | 0.14 | 0.09 |
| | **350M** | −0.00 | 0.01 | 0.02 | 0.03 | 0.12 | 0.14 | 0.09 |
| | **1.3B** | −0.00 | 0.01 | 0.02 | 0.03 | 0.12 | 0.14 | 0.09 |
| | **2.7B** | −0.00 | 0.01 | 0.02 | 0.03 | 0.12 | 0.14 | 0.09 |
| | **6.7B** | −0.00 | 0.01 | 0.02 | 0.03 | 0.12 | 0.14 | 0.09 |
| | **13B** | −0.04 | −0.02 | −0.01 | −0.00 | 0.09 | 0.11 | 0.05 |
| | **30B** | −0.11 | −0.09 | −0.08 | −0.08 | 0.02 | 0.03 | −0.02 |

(a) RTE

| | | **FT** | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | **125M** | **350M** | **1.3B** | **2.7B** | **6.7B** | **13B** | **30B** |
| **ICL** | **125M** | −0.00 | 0.00 | 0.02 | 0.01 | 0.10 | 0.11 | 0.07 |
| | **350M** | −0.00 | 0.00 | 0.02 | 0.01 | 0.10 | 0.11 | 0.07 |
| | **1.3B** | −0.01 | −0.00 | 0.01 | 0.01 | 0.10 | 0.11 | 0.07 |
| | **2.7B** | −0.01 | −0.00 | 0.01 | 0.01 | 0.09 | 0.10 | 0.07 |
| | **6.7B** | −0.01 | −0.01 | 0.01 | 0.00 | 0.09 | 0.10 | 0.06 |
| | **13B** | −0.03 | −0.03 | −0.02 | −0.02 | 0.07 | 0.08 | 0.04 |
| | **30B** | −0.07 | −0.07 | −0.05 | −0.06 | 0.03 | 0.04 | 0.00 |

(b) MNLI

Table 1: Difference between average **out-of-domain performance** of ICL and FT on RTE (a) and MNLI (b) across model sizes. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to in-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference.

is less clear and most models perform worse than the majority baseline. Table 6 (in Appendix A.4) compares our ICL results with previous work.

For FT, we similarly observe that in-domain performance increases with model size. Moreover, across all datasets and model sizes, FT with just 16 examples leads to similar in-domain performance as ICL (see Tables 8 and 9 in Appendix B.1 for statistical tests comparing in-domain performance of FT and ICL on RTE and MNLI). On QQP, we again observe no clear relationship between model size and performance. Only 10 out of 70 models perform better than the majority baseline.

**Out-of-domain performance** Turning to OOD performance, we find that for MNLI and QQP most of the ICL models perform close to the majority baseline. On MNLI, only the largest model (30B) shows good OOD generalization for 4 out of 10 runs. On RTE, in-domain and OOD performance of the 30B model mostly overlap, which is consistent with the findings of Si et al. (2023). In particular, when comparing the relationship between the in-domain and OOD performance of the 30B model to the smallest fine-tuned models (125M and 350M) one might conclude that ICL leads to better OOD performance; for FT on MNLI and RTE, indeed, the smallest models have poor OOD performance.

However, as model size increases, OOD performance increases as well, demonstrating that even in the challenging few-shot setting, fine-tuned models can generalize OOD. Focusing on the largest

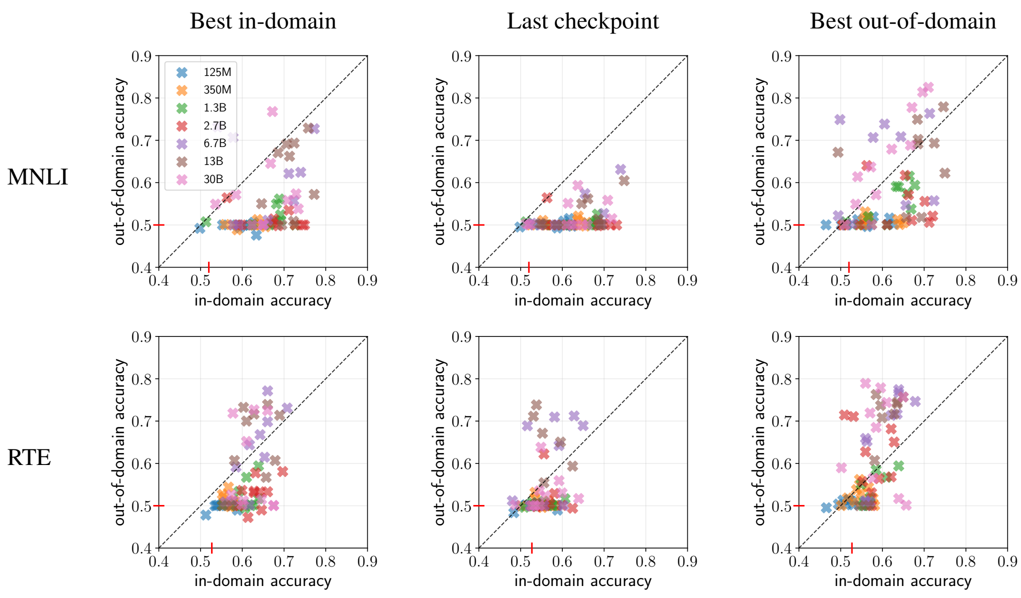


Figure 3: Comparing model selection strategies in FT. The first and second rows show results for MNLI and RTE respectively. We train on 16 examples and plot results for 10 runs for each model size. — in the x- and y-axes indicates majority class accuracy.

models (6.7B, 13B, and 30B) fine-tuned on MNLI, we find that for most runs, OOD performance is on par or even better than in-domain performance. On RTE, the trend is even stronger. Even with the 1.3B model, we observe good in-domain and OOD performance, and both improve as the models get larger. Notably, for many models, OOD performance is even better than in-domain performance.

In summary, **our comparison shows that fine-tuned language models can generalize OOD as well or even better than models adapted via ICL** (see statistical tests comparing them in Table 1). This highlights the importance of comparing adaptation approaches using models of the same size.

### 4.1 A closer look at FT generalization

Having established that few-shot FT can also lead to strong in-domain and OOD performance, we now focus on better understanding the individual choices that impact the in-domain and out-of-domain performance of FT. Given that on QQP, most models achieve close to majority accuracy, we focus on MNLI and RTE in the following and present results for QQP in Appendix B.

**The role of model selection** Our FT results in Figure 2 show that many fine-tuned models lead to good out-of-domain generalization. But what is the role of model selection in identifying these checkpoints? To answer this question, we compare selecting the model (a) with the best in-domain performance, (b) at the end of fine-tuning, and (c) with the best out-of-domain performance. Figure 3 shows the results when fine-tuning on 16 examples. Results for additional sample sizes are shown in Figures 11 to 13 in Appendix B.3.

Our results show that when performing model selection according to in-domain performance, only the largest models achieve good OOD performance. On the other hand, when performing model selection according to OOD performance, smaller models can also generalize well (e.g. for the 2.7B model on RTE, 7 out of 10 models have equal or even better OOD than in-domain performance), and this trend persists as model size increases. Interestingly, on RTE, we also observe models with a strong OOD performance when selecting the last checkpoint, which typically leads to poor OOD performance on MNLI.

**Training on more data** In contrast to ICL, where the maximum number of demonstrations is limited by the context size of a model, FT allows us to perform task adaptation using arbitrary amounts of data. Here, we analyze how the relationship between in-domain and OOD performance is impacted by training on more data. Figure 4 shows the results for MNLI and RTE, and results for QQP are provided in Figure 13 in Appendix B.3. For the smallest models, we find that while in-domain performance increases with more training data, OOD performance remains low, which is consistent with

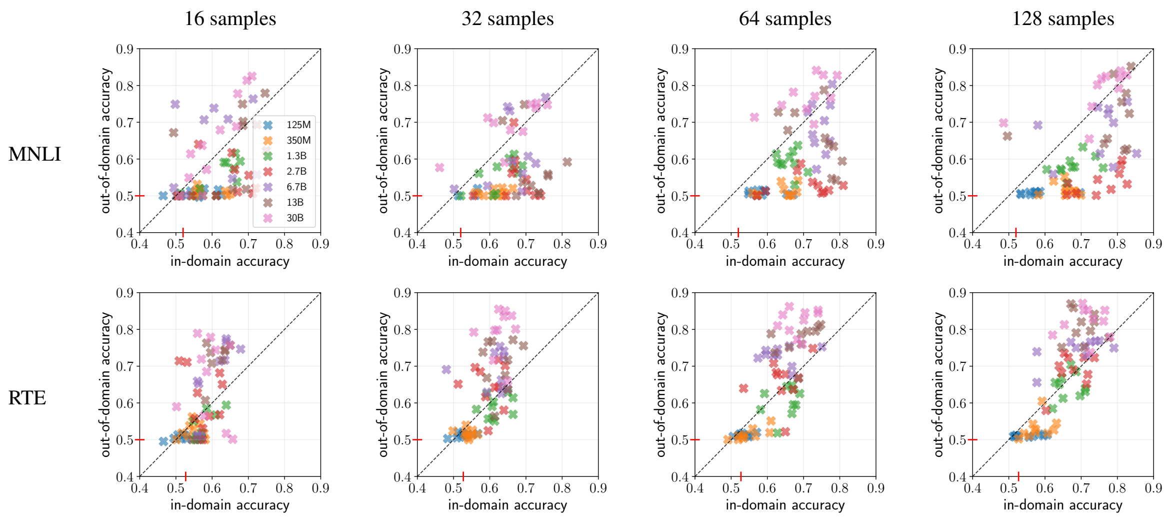


Figure 4: Exploring the effect of increasing training examples on FT. The first and second rows show results for MNLI and RTE respectively. We plot results for 10 runs for each model size and perform model selection according to out-of-domain performance. — in the x- and y-axes indicates majority class accuracy.

previous work (Utama et al., 2021). However, for larger models, OOD performance improves as the amount of training data increases and the same trend can be observed when performing model selection according to in-domain performance (see Figures 11 to 13 in Appendix B.3).

**How much OOD data is needed?** In the experiments so far, we evaluated the models on the full evaluation set (unless mentioned otherwise). Further, we selected FT models based on this evaluation; choosing the best model according to its in-domain or OOD performance in this entire set. This setup is not realistic, since in such a scenario where large amounts of data are available for evaluation, it can be used more effectively for training (Zhu et al., 2023). Hence, in this experiment, we quantify the ability to estimate a model's performance on OOD data using smaller evaluation sets. We fine-tune OPT 13B on MNLI using 128 examples using three different data seeds and plot the OOD generalization in Figure 5. Our results show that using just 50 randomly selected examples is sufficient to distinguish checkpoints that generalize well from those that do not, which would allow us to select, with only these 50 examples, the best OOD checkpoint in a model's training run. This is also reflected in the Pearson correlation of the OOD performance during FT when evaluating it on all vs. 50 examples, which is very high: 0.99.

## 4.2 Comparing fine-tuning approaches

Lastly, we investigate the importance of performing pattern-based FT instead of vanilla FT by fine-tuning a model with a randomly initialized classification head (Howard and Ruder, 2018; Devlin et al., 2019). Further, as an extra fine-tuning strategy, we also apply LoRA (Hu et al., 2022) – a recently proposed approach for parameter-efficient fine-tuning – on top of pattern-based FT for comparison. This makes adaptation via FT more similar to adaptation via ICL as it allows the re-use of a large fraction of the weights of a pre-trained language model across tasks.[15] We fine-tune all models on 16 examples from RTE and present the results in Figure 6. For all FT approaches, we observe a clear improvement in both in-domain and OOD performance as models become larger. Compared to vanilla FT, pattern-based FT leads to better overall performance. When combined with

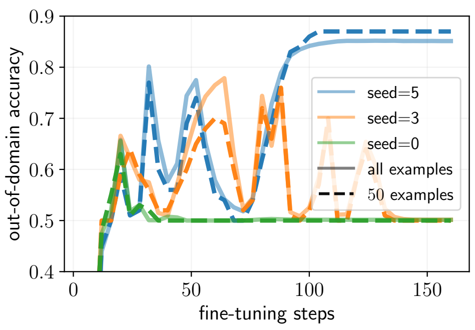


Figure 5: Estimating OOD performance using less data. We compare OOD performance estimated using all vs. 50 examples when fine-tuning OPT 13B on RTE. Each color corresponds to a run with a different data seed.

[15]We provide more details on both approaches in Appendix A.5.

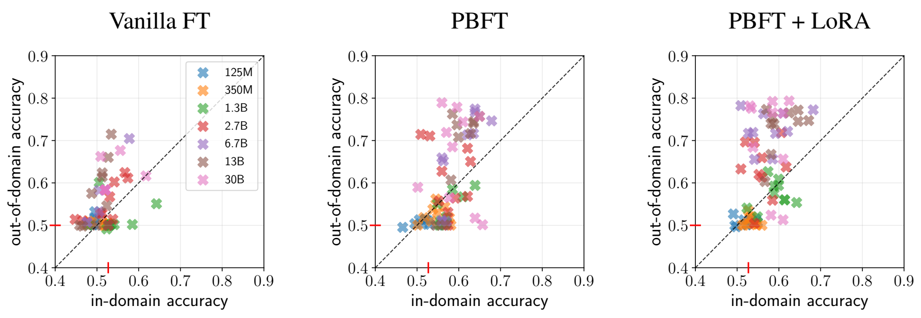


Figure 6: Comparing FT approaches on RTE. We use 16 examples and perform model selection according to out-of-domain performance. — in the x- and y-axes indicates majority class accuracy.

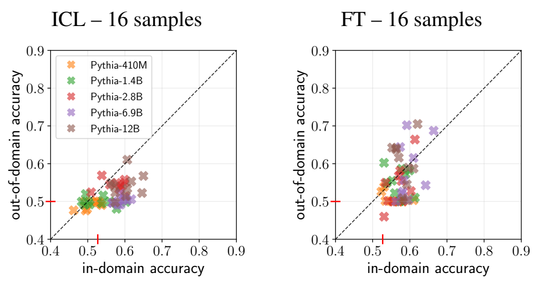


Figure 7: ICL and FT results for Pythia models on RTE. We fine-tune models using PBFT. For each approach, we use 16 examples and perform model selection according to in-domain performance. We plot 10 runs per model size which differ only in the data seed. — in the x- and y-axes indicates majority class accuracy.

|  |  | FT |  |  |  |  |
|---|---|---|---|---|---|---|
|  |  | **410M** | **1.4B** | **2.8B** | **6.9B** | **12B** |
| **ICL** | **410M** | 0.02 | 0.06 | 0.05 | 0.09 | 0.11 |
|  | **1.4B** | 0.01 | 0.05 | 0.04 | 0.08 | 0.10 |
|  | **2.8B** | −0.03 | 0.01 | −0.00 | 0.04 | 0.06 |
|  | **6.9B** | 0.01 | 0.05 | 0.04 | 0.08 | 0.10 |
|  | **12B** | −0.03 | 0.01 | −0.00 | 0.04 | 0.06 |

Table 2: Difference between average **out-of-domain performance** of ICL and FT with Pythia models on RTE. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to in-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference.

LoRA, pattern-based FT leads to very similar performance as training all parameters. These results demonstrate the generality of our findings beyond a specific FT method.

### 4.3 Our findings generalize beyond OPT

Figure 7 provides a comparison of ICL and FT using Pythia models[16] of different sizes ranging from 410M to 12B parameters (Biderman et al., 2023). The corresponding significance tests for OOD performance are shown in Table 2 (significance tests for in-domain performance are in Appendix C). Similar to OPT, all Pythia models have been trained on the same data, and in the same order. We fine-tune using PBFT and select models according to in-domain performance. The results for additional patterns, model selection strategies, and sample sizes are discussed in Appendix C.

Similarly to OPT, we observe a clear effect of model size on both in-domain and OOD performance. For most model sizes, FT leads to significantly better OOD performance than ICL and both the in-domain and OOD performance of Pythia models improve drastically as we fine-tune on more data (see Figure 16). This demonstrates the generality of our findings beyond a single model.

## 5 Discussion

Our findings in the previous section demonstrate that fine-tuned language models can generalize OOD too, highlighting the importance of comparing adaptation approaches fairly. In this section, we present further insights from our experiments and provide a high-level comparison of the pros and cons of adaptation via ICL and FT.

**What signal to learn from?** Both our ICL and FT results exhibit a large variance in both in-domain and OOD performance. Our results show different OOD behavior during FT when varying only the data seed. In addition, as previous work has shown, the choice of patterns and ver-

[16]We use the non-deduped models.

balizers impact both ICL and PBFT performance in unintuitive ways. For instance, Webson and Pavlick (2022) find that pattern-based fine-tuned models perform well even when using misleading patterns. Here, we find that ICL's generalization is heavily dependent on the choice of pattern and verbalizer. This shows the importance of the choice of training data and patterns for task adaptation.

**Advances in task adaptation** The success of ICL led to the development of new methods for improving on it further, such as calibration (Zhao et al., 2021), and chain-of-thought prompting (Wei et al., 2022). In this work, we focus on the 'vanilla' version of ICL and the fine-tuning approach most similar to it – pattern-based fine-tuning. Our results suggest that these two approaches are more similar than previously thought, as they achieve similar performance both in-domain and OOD. As such, new methods for ICL can also be applied to PBFT, and we expect them to achieve similar results.

**Analyzing the fine-tuning loss surface** Looking at the OOD generalization curves throughout fine-tuning (in Figure 5 and additional plots in Appendix D), we observe that for some runs, OOD performance fluctuates heavily and models change their generalization 'strategy' during FT. In Figure 5, we can see that some fine-tuning runs undergo a dramatic change in OOD performance after 75 steps. We leave it to future work to further study this behavior and the relationship between the FT loss surface and OOD generalization (Shwartz-Ziv et al., 2022; Juneja et al., 2023).

## 6 Comparing FT and ICL

This section examines the key features for task adaptation and compares FT and ICL. We summarize our findings in Table 3. We begin by discussing features related to user interaction, which can be found in the first part of the table. FT requires expertise in model training, whereas ICL only requires natural language, i.e., non-experts can use this approach more easily. ICL is also highly reusable as it does not modify the pre-trained model and hence, the same model can be used for many tasks; FT, however, is not as reusable (with the exception of parameter-efficient methods) and typically results in a specialized model per task. Unfortunately, despite its user-friendliness and reusability, ICL does not work out of the box for some tasks which require more sophisticated

| **Feature** | **FT** | **ICL** |
|---|---|---|
| Users | Experts | Experts & Non-experts |
| Interaction | Pre-defined | Textual |
| Reusability | Medium | High |
| Applicability to low-resource languages | High | Limited |
| Requires training | Yes | No |
| Inference time | \|test example\| | \|test example\| + \|demonstrations\| |
| \|Demonstrations\| | Unlimited | $\leq$100 |
| Variance | High | High |
| SOTA | Yes | Yes |
| Size scaling | Standard | Standard |
| \|Demonstrations\| scaling | Standard | Limited |
| Invented | 2018 | 2020 |
| Well understood | No | No |

Table 3: A high-level comparison between key features of fine-tuning and in-context learning.

prompting (Wei et al., 2022).

ICL requires large models to work in contrast to FT, which works well even with small models (Devlin et al., 2019). This hinders the applicability of ICL to models developed for low-resource languages, as training billion parameter-scale models requires huge amounts of training data, which are simply unavailable for many languages. As such, FT is still the dominating adaptation approach in this setting (Pfeiffer et al., 2022; Alabi et al., 2022, *inter alia*).

Next, we compare technical details regarding the training and inference of such approaches. While FT requires training (which when dealing with large models can become expensive), ICL does not. On the other hand, the inference time of fine-tuned models is much smaller than ICL, since it only includes the time that it takes to process the minimal pattern and the test instance. When using ICL, each test instance has to include all of the demonstrations as well, which increases the inference time. The fixed context size of the model also limits the number of demonstrations that can be used[17], while FT allows for unlimited training examples. We show in this work that both methods can achieve strong performance on both in-domain and OOD datasets. Both approaches improve with model size, but FT benefits more from additional samples than ICL does, as was also shown in previous work (Min et al., 2022).

Finally, we highlight that both methods are

[17]Note that some methods allow an infinite context (e.g. Press et al., 2022a; Martins et al., 2022). Most current successful LMs, however, have limited context sizes.

relatively recent: vanilla FT was invented in 2018 (Howard and Ruder, 2018) and ICL in 2020 (Brown et al., 2020).[18] As such, these methods are still poorly understood, and more research is required on both approaches to better understand their strengths and weaknesses.

## 7 Related work

Brown et al. (2020) compare GPT-3's few-shot in-context learning performance with fine-tuned language models trained in the fully supervised setting, finding that both approaches lead to similar results in question answering. However, the fine-tuned models they compare ICL to are smaller models, making the task adaptation comparison unfair. For SuperGLUE, while using smaller models, they find that FT largely outperforms ICL. This is consistent with our findings. Even in the few-shot setting, fine-tuned language models can outperform ICL when comparing models of the same size. Recently, Liu et al. (2022) compared parameter-efficient few-shot FT of T0 (Sanh et al., 2022) to ICL with GPT-3, finding that their parameter-efficient FT approach outperforms ICL. This is consistent with our findings; however, unlike our work, they only consider in-domain performance.

Focusing on OOD performance, Si et al. (2023) investigate the generalization of GPT-3 along various axes, including generalization under covariate shift – as we do. However, they compare models of different sizes, i.e., RoBERTa-large and GPT-3 (which has 500 times the number of parameters), and different training settings, i.e., fully supervised for FT vs. few-shot for ICL. They observe much better OOD performance for ICL than FT, concluding that ICL with GPT-3 is more robust than FT using BERT or RoBERTa. While this conclusion is valid, it holds for specific models, rather than the methods in general. We show how important it is to compare methods fairly. Based on our comparable results, fine-tuning language models results in similar or even better OOD generalization. Another work that compares the OOD generalization of different adaptation approaches is Awadalla et al. (2022). Unlike our choice of MNLI and RTE, they investigate the robustness of question answering models under various types of distribution shifts and find that ICL is more robust to distribution shifts than FT. Moreover, they argue that for FT, increasing model size does not have a strong impact on generalization. However, they don't scale beyond 1.5B parameters. Our findings suggest that the relationship between in-domain and OOD performance does depend on model size.

While we focus on the task adaptation of decoder-only models, Utama et al. (2021) investigate the OOD generalization of encoder-only models adapted via pattern-based few-shot FT. For MNLI and HANS, they find that these models adopt similar inference heuristics to those trained with vanilla FT and hence perform poorly OOD. They observe that models rely even more on heuristics when fine-tuned on more data. This is in contrast to our results where we find that pattern-based few-shot FT can lead to good OOD generalization, and OOD generalization improves as we train on more data. We attribute this to the fact that they experiment with a smaller model (RoBERTa-large; 350M).[19] Lastly, Bandel et al. (2022) show that masked language models can generalize well on HANS if fine-tuned for a sufficient number of steps. While they focus on fine-tuning on the entire dataset, their findings provide additional evidence that fine-tuned language models can generalize well OOD.

## 8 Conclusion

We perform a fair comparison between in-domain and OOD generalization of two alternative task adaptation strategies: Few-shot ICL and FT. We compare OPT models (Zhang et al., 2022) ranging from 125M to 30B parameters on three classification datasets across two tasks. We find that for both approaches, performance improves as models become larger. For the largest models we experiment with (OPT-30B), we find that FT outperforms ICL on both in-domain and OOD performance and even improves further as we train on more data. However, our results also demonstrate that the performance of both FT and ICL exhibits high variance, highlighting that truly robust task adaptation remains an open challenge. We end by providing a high-level comparison between the two approaches, listing the benefits and limitations of each, and discussing some future directions.

[18]PBFT was also invented in 2020 (Schick and Schütze, 2021).

[19]This is also related to Warstadt et al.'s (2020) results, who show that better pre-trained models are less prone to rely on superficial (and potentially spurious) features for predictions.

## 9 Limitations

In this work, we focus on a specific type of OOD generalization, namely, covariate shift (Hupkes et al., 2022). Under this setup, we refer to OOD as the specific challenge datasets we use. As such, different conclusions might be reached by repeating the experiments and evaluating different datasets.

We focus specifically on OPT decoder-only models as the goal of our work is to compare the generalization of adaptation via fine-tuning vs. in-context learning using the same pre-trained model. To the best of our knowledge, existing encoder-only models do not have strong in-context learning abilities. For encoder–decoder models such as T5, only recent variants such as Flan-T5 (Chung et al., 2022) demonstrate the ability to respond well to instructions. However, these models require an additional supervised fine-tuning step on instruction data. This makes it challenging to attribute generalization abilities (or the lack thereof) to specific adaptation techniques (fine-tuning vs in-context learning). Hence, we focus on decoder-only models pre-trained exclusively with a language modeling objective.

Many recent papers that experiment with in-context learning use GPT-3. While fine-tuning GPT-3 is possible via an API, it is unclear what fine-tuning approach is used behind that API. Since this makes a fair comparison difficult, we chose not to experiment with GPT-3.

While similarly large models (e.g. OPT-175B) are publicly available, we do not have the computational resources to run such models. While we expect the trends we observe in this work to hold with larger models, we are not able to empirically test that. Moreover, we only experiment with English language models as, to the best of our knowledge, there are no publicly available models which are similar to OPT (decoder-only models of various sizes trained on the same data) for other languages.

Finally, we only experiment with basic FT and ICL methods. However, for both approaches there exist more advanced techniques which we do not consider (e.g. calibration; Zhao et al., 2021). We note that such techniques can typically be applied for both adaptation approaches. Hence we expect an improvement for one method to improve the other as well.

## Acknowledgments

We are grateful to Vagrant Gautam for their valuable feedback and patience when proofreading our work. We also thank Badr Abdullah for his help with proofreading and feedback during early stages of this work. Marius Mosbach acknowledges funding from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 232722074 – SFB 1102.

## References

Jesujoba O. Alabi, David Ifeoluwa Adelani, Marius Mosbach, and Dietrich Klakow. 2022. Adapting pre-trained language models to African languages via multilingual adaptive fine-tuning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4336–4349, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Anas Awadalla, Mitchell Wortsman, Gabriel Ilharco, Sewon Min, Ian Magnusson, Hannaneh Hajishirzi, and Ludwig Schmidt. 2022. Exploring the landscape of distributional robustness for question answering models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5971–5987, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Elron Bandel, Yoav Goldberg, and Yanai Elazar. 2022. Lexical generalization improves with larger models and longer training. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4398–4410, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. *arXiv preprint arXiv:2304.01373*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens

Winter, Chris 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*, volume 33, pages 1877–1901. Curran Associates, Inc.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, Berlin, Heidelberg. Springer Berlin Heidelberg.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

Yanai Elazar, Hongming Zhang, Yoav Goldberg, and Dan Roth. 2021. Back to square one: Artifact detection, training and commonsense disentanglement in the Winograd schema. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10486–10500, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2021a. A framework for few-shot language model evaluation. Zenodo.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021b. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Dieuwke Hupkes, Mario Giulianelli, Verna Dankers, Mikel Artetxe, Yanai Elazar, Tiago Pimentel, Christos Christodoulopoulos, Karim Lasri, Naomi Saphra, Arabella Sinclair, Dennis Ulmer, Florian Schottmann, Khuyagbaatar Batsuren, Kaiser Sun, Koustuv Sinha, Leila Khalatbari, Maria Ryskina, Rita Frieske, Ryan Cotterell, and Zhijing Jin. 2022. State-of-the-art generalisation research in NLP: A taxonomy and review. *arXiv preprint arXiv:2210.03050*.

Jeevesh Juneja, Rachit Bansal, Kyunghyun Cho, João Sedoc, and Naomi Saphra. 2023. Linear connectivity reveals generalization strategies. In *The Eleventh International Conference on Learning Representations*.

Haokun Liu, Derek Tam, Muqeeth Mohammed, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems*.

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

Robert Logan IV, Ivana Balazevic, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2022. Cutting down on prompts and parameters: Simple few-shot learning with language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2824–2835, Dublin, Ireland. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *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.

Pedro Henrique Martins, Zita Marinho, and Andre Martins. 2022. $\infty$-former: Infinite memory transformer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5468–5485, Dublin, Ireland. Association for Computational Linguistics.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*.

OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Jonas Pfeiffer, Naman Goyal, Xi Lin, Xian Li, James Cross, Sebastian Riedel, and Mikel Artetxe. 2022. Lifting the curse of multilinguality by pre-training modular transformers. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3479–3495, Seattle, United States. Association for Computational Linguistics.

Ofir Press, Noah Smith, and Mike Lewis. 2022a. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2022b. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. DeepSpeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5569–5578, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Lakshay Sharma, Laura Graesser, Nikita Nangia, and Utku Evci. 2019. Natural language understanding with the Quora question pairs dataset. *arXiv preprint arXiv:1907.01041*.

Ravid Shwartz-Ziv, Micah Goldblum, Hossein Souri, Sanyam Kapoor, Chen Zhu, Yann LeCun, and Andrew Gordon Wilson. 2022. Pre-train your loss: Easy Bayesian transfer learning with informative priors. In *Advances in Neural Information Processing Systems*.

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Lee Boyd-Graber, and Lijuan Wang. 2023. Prompting GPT-3 to be reliable. In *The Eleventh International Conference on Learning Representations*.

Derek Tam, Rakesh R. Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4980–4991, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Prasetya Utama, Nafise Sadat Moosavi, Victor Sanh, and Iryna Gurevych. 2021. Avoiding inference heuristics in few-shot prompt-based finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9063–9074, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman. 2020. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 217–235, Online. Association for Computational Linguistics.

Albert Webson and Ellie Pavlick. 2022. Do prompt-based models really understand the meaning of their prompts? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2300–2344, Seattle, United States. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Dawei Zhu, Xiaoyu Shen, Marius Mosbach, Andreas Stephan, and Dietrich Klakow. 2023. Weaker than you think: A critical look at weakly supervised learning. In *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada. Association for Computational Linguistics.

## A Experimental details

We access all models via huggingface transformers (Wolf et al., 2020) and use its DeepSpeed (Rasley et al., 2020) integration for efficient distributed training and evaluation.

### A.1 Hardware

We run our experiments on 8x A100 GPUs with 80GB of memory.

### A.2 Label distribution

Table 4 shows the accuracy of the majority class label on each of the datasets. Note that MNLI (when merging the neutral and contradiction classes) and PAWS-QQP are highly unbalanced.

| Dataset | Majority class |
|---|---|
| MNLI (remove neutral) | 0.512 |
| MNLI (merge neutral and contradiction) | 0.645 |
| RTE | 0.527 |
| QQP | 0.632 |
| HANS | 0.500 |
| PAWS-QQP | 0.718 |

Table 4: Accuracy of the majority class label for each dataset.

### A.3 In-context learning: Additional details

**Patterns** We present the patterns used for ICL in Table 5. We obtain the `GPT-3` pattern from Brown et al. (2020). The `eval-harness` pattern is based on Gao et al. (2021a).

| Dataset(s) | Pattern | Text | Answer prefix | Target tokens |
|---|---|---|---|---|
| MNLI, RTE | `minimal` | {premise} {hypothesis} ? | – | Yes, No |
| MNLI, RTE | `gpt-3` | {premise} question: {hypothesis} Yes or No? | answer: | Yes, No |
| MNLI, RTE | `eval-harness` | {premise} \n Question: {hypothesis} True or False? | \n Answer: | True, False |
| QQP | `minimal` | {question 1} {question 2} ? | – | Yes, No |
| QQP | `eval-harness` | Question 1: {question 1} \n Question 2:{question 2} \n Question: Do both questions ask the same thing? | Answer: | Yes, No |

Table 5: Patterns used for ICL. The minimal patterns are used for PBFT as well.

### A.4 In-context learning: Comparison with previous work

Table 6 compares our ICL results to results from previous work. On RTE and MNLI we achieve comparable performance to previous work. On QQP, our ICL results are much worse (and very close to the majority class classifier). We hypothesize that this is due to the difference in model size (GPT-3 with 175B parameters vs. OPT with 30B parameters) and hence focus on MNLI and RTE for most of our experiments.

### A.5 Fine-tuning: Additional details

**Vanilla FT** Vanilla FT (Howard and Ruder, 2018; Devlin et al., 2019) is one of the most commonly used task adaptation approaches for pre-trained language models. During FT we typically: (i) replace the model's language modeling head with a new randomly initialized classification head; (ii) update all model parameters, as well as the new head's, on the downstream task's training data.[20] When trained on entire datasets, fine-tuned language models dominate academic leaderboards, such as GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019). However, despite their strong in-domain performance, fine-tuned

[20]We will refer to any FT approach that uses a randomly initialized classifier as vanilla FT.

| Model | Dataset | In-domain | Out-of-domain |
|---|---|---|---|
| GPT-3 175B | MNLI | 77.6 | 75.3 |
| OPT 30B | RTE | 62.0 | – |
| GPT-3 175B | QQP | 83.5 | 73.7 |
| OPT 30B | MNLI | 71.4 (74.9) | 56.7 (72.3) |
| OPT 30B | RTE | 61.7 (66.8) | 60.5 (65.4) |
| OPT 30B | QQP | 42.0 (63.1) | 49.8 (53.3) |

Table 6: Comparing ICL results from previous work (first three rows) with ours (last three rows). In our results we report average and maximum performance (in parentheses) of the largest model. Previous results are from Si et al. (2023) for GPT-3 and Zhang et al. (2022) for OPT.

language models tend to generalize poorly OOD, which is often attributed to adopting inference heuristics during FT (McCoy et al., 2019; Elazar et al., 2021).

**Parameter-efficient FT** Parameter-efficient FT methods update only a small number of parameters relative to the total number of parameters of the pre-trained model (Houlsby et al., 2019; Ben Zaken et al., 2022; Hu et al., 2022, *inter alia*). Such approaches can be applied to either vanilla or prompt-based FT and are appealing since they allow large parts of a model to be re-used across tasks.

| Hyperparameter | Value |
|---|---|
| Optimizer | AdamW |
| Learning rate | $10^{-5}$ |
| Learning rate schedule | linear warmup then constant |
| Warmup ratio | 10% of total steps |
| Weight decay | 0.0 |
| Dropout | 0.1 |
| Batch size | 32 |
| Epochs | 40 |
| Total steps | $\frac{\text{\#samples}}{\text{batch size}} * \text{epochs}$ |

Table 7: FT hyperparameters.

**Hyperparameters** Table 7 provides an overview of all hyperparameters used during FT.

## B Additional results for OPT models

### B.1 Significance tests

Tables 1 and 8 to 10 show the results of a Welch's t-test comparing the average in-domain and out-of-domain performance of ICL and PBFT on RTE and MNLI. We use 16 samples and 10 different seeds for each approach and consider a p-value of 0.05 to be statistically significant. For FT, we compare two different approaches of model selection: (1) based on in-domain performance and (2) based on out-of-domain performance (note that these are the same models as those shown in the first row of Figure 1).

For RTE, our results show that ICL outperforms FT only when comparing large models to smaller models. However, when comparing models of the same size, FT performs at least equally well to ICL, and in some cases even significantly better. For MNLI, for larger models (6.7B onwards) ICL outperforms FT in terms of in-domain performance. Looking at OOD performance, however, we again see that ICL only outperforms FT when comparing large models to much smaller models.

| | FT | | | | | | |
|---|---|---|---|---|---|---|---|
| ICL | **125M** | **350M** | **1.3B** | **2.7B** | **6.7B** | **13B** | **30B** |
| **125M** | 0.03 | 0.04 | 0.08 | 0.11 | 0.10 | 0.09 | 0.10 |
| **350M** | 0.03 | 0.05 | 0.08 | 0.11 | 0.10 | 0.10 | 0.10 |
| **1.3B** | 0.03 | 0.04 | 0.08 | 0.10 | 0.10 | 0.09 | 0.09 |
| **2.7B** | 0.00 | 0.02 | 0.05 | 0.08 | 0.07 | 0.07 | 0.07 |
| **6.7B** | −0.06 | −0.04 | −0.01 | 0.02 | 0.01 | 0.01 | 0.01 |
| **13B** | −0.06 | −0.05 | −0.01 | 0.02 | 0.01 | 0.00 | 0.01 |
| **30B** | −0.06 | −0.04 | −0.01 | 0.02 | 0.01 | 0.01 | 0.01 |

(a) RTE

| | FT | | | | | | |
|---|---|---|---|---|---|---|---|
| ICL | **125M** | **350M** | **1.3B** | **2.7B** | **6.7B** | **13B** | **30B** |
| **125M** | 0.07 | 0.09 | 0.13 | 0.14 | 0.12 | 0.17 | 0.13 |
| **350M** | 0.05 | 0.07 | 0.11 | 0.13 | 0.11 | 0.16 | 0.11 |
| **1.3B** | −0.02 | −0.00 | 0.03 | 0.05 | 0.03 | 0.08 | 0.03 |
| **2.7B** | 0.01 | 0.03 | 0.07 | 0.08 | 0.06 | 0.11 | 0.06 |
| **6.7B** | −0.06 | −0.04 | −0.00 | 0.01 | −0.01 | 0.04 | −0.00 |
| **13B** | −0.13 | −0.11 | −0.07 | −0.06 | −0.08 | −0.03 | −0.08 |
| **30B** | −0.11 | −0.09 | −0.05 | −0.04 | −0.06 | −0.01 | −0.06 |

(b) MNLI

Table 8: Difference between average **in-domain performance** of ICL and FT on RTE (a) and MNLI (b) across model sizes. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to in-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.

| | FT | | | | | | |
|---|---|---|---|---|---|---|---|
| ICL | **125M** | **350M** | **1.3B** | **2.7B** | **6.7B** | **13B** | **30B** |
| **125M** | −0.01 | 0.02 | 0.05 | 0.05 | 0.07 | 0.07 | 0.07 |
| **350M** | −0.01 | 0.02 | 0.05 | 0.05 | 0.08 | 0.08 | 0.07 |
| **1.3B** | −0.01 | 0.01 | 0.04 | 0.04 | 0.07 | 0.07 | 0.06 |
| **2.7B** | −0.04 | −0.01 | 0.02 | 0.02 | 0.05 | 0.05 | 0.04 |
| **6.7B** | −0.09 | −0.07 | −0.04 | −0.04 | −0.01 | −0.01 | −0.02 |
| **13B** | −0.10 | −0.07 | −0.04 | −0.04 | −0.02 | −0.02 | −0.02 |
| **30B** | −0.10 | −0.07 | −0.04 | −0.04 | −0.01 | −0.01 | −0.02 |

(a) RTE

| | FT | | | | | | |
|---|---|---|---|---|---|---|---|
| ICL | **125M** | **350M** | **1.3B** | **2.7B** | **6.7B** | **13B** | **30B** |
| **125M** | 0.03 | 0.05 | 0.09 | 0.10 | 0.07 | 0.13 | 0.08 |
| **350M** | 0.01 | 0.03 | 0.07 | 0.09 | 0.05 | 0.12 | 0.06 |
| **1.3B** | −0.07 | −0.04 | −0.01 | 0.01 | −0.02 | 0.04 | −0.01 |
| **2.7B** | −0.03 | −0.01 | 0.02 | 0.04 | 0.01 | 0.07 | 0.02 |
| **6.7B** | −0.10 | −0.08 | −0.04 | −0.03 | −0.06 | 0.00 | −0.05 |
| **13B** | −0.17 | −0.15 | −0.11 | −0.10 | −0.13 | −0.07 | −0.12 |
| **30B** | −0.16 | −0.13 | −0.10 | −0.08 | −0.11 | −0.05 | −0.10 |

(b) MNLI

Table 9: Difference between average **in-domain performance** of ICL and FT on RTE (a) and MNLI (b) across model sizes. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to out-of-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.

| | FT | | | | | | |
|---|---|---|---|---|---|---|---|
| ICL | **125M** | **350M** | **1.3B** | **2.7B** | **6.7B** | **13B** | **30B** |
| **125M** | 0.01 | 0.02 | 0.03 | 0.11 | 0.16 | 0.18 | 0.16 |
| **350M** | 0.01 | 0.02 | 0.03 | 0.11 | 0.16 | 0.18 | 0.16 |
| **1.3B** | 0.01 | 0.02 | 0.03 | 0.11 | 0.16 | 0.18 | 0.16 |
| **2.7B** | 0.00 | 0.02 | 0.03 | 0.11 | 0.15 | 0.18 | 0.16 |
| **6.7B** | 0.01 | 0.02 | 0.03 | 0.11 | 0.15 | 0.18 | 0.16 |
| **13B** | −0.03 | −0.01 | 0.00 | 0.08 | 0.12 | 0.14 | 0.13 |
| **30B** | −0.10 | −0.08 | −0.07 | 0.01 | 0.05 | 0.07 | 0.06 |

(a) RTE

| | FT | | | | | | |
|---|---|---|---|---|---|---|---|
| ICL | **125M** | **350M** | **1.3B** | **2.7B** | **6.7B** | **13B** | **30B** |
| **125M** | 0.00 | 0.01 | 0.05 | 0.04 | 0.13 | 0.14 | 0.17 |
| **350M** | 0.00 | 0.01 | 0.05 | 0.04 | 0.13 | 0.14 | 0.17 |
| **1.3B** | 0.00 | 0.01 | 0.05 | 0.04 | 0.13 | 0.14 | 0.16 |
| **2.7B** | 0.00 | 0.01 | 0.05 | 0.04 | 0.13 | 0.14 | 0.16 |
| **6.7B** | −0.01 | −0.00 | 0.04 | 0.03 | 0.12 | 0.13 | 0.16 |
| **13B** | −0.03 | −0.02 | 0.02 | 0.01 | 0.10 | 0.11 | 0.13 |
| **30B** | −0.06 | −0.06 | −0.01 | −0.02 | 0.06 | 0.08 | 0.10 |

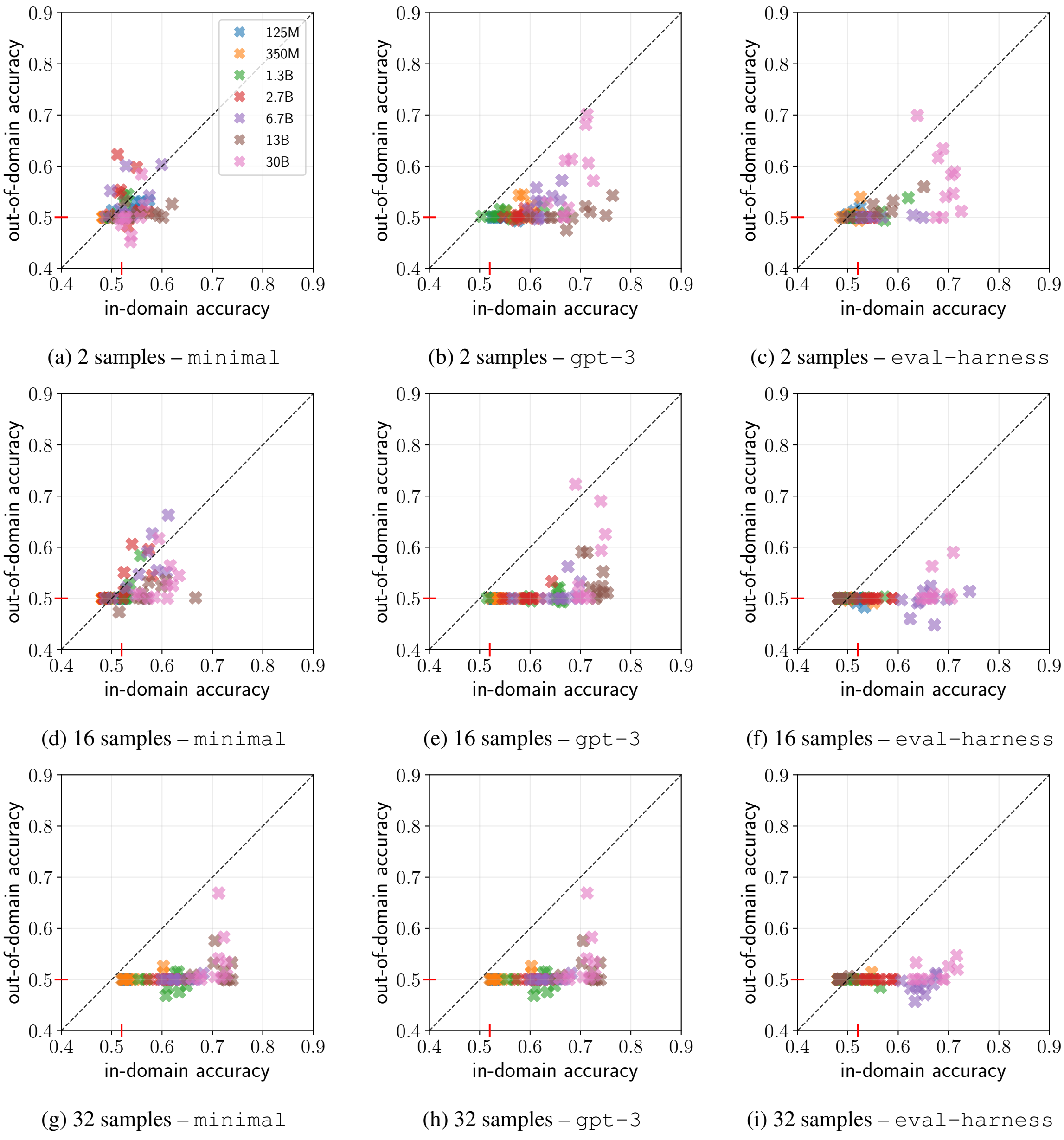
(b) MNLI

Table 10: Difference between average **out-of-domain performance** of ICL and FT on RTE (a) and MNLI (b) across model sizes. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to out-of-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.

## B.2 In-context learning

Figures 8, 9, and 10 show ICL results on MNLI, RTE, and QQP for all OPT model sizes grouped by number of demonstrations and patterns.

**Sensitivity to pattern choice and number of examples** On MNLI and RTE, we find that only the largest model benefits from the instructive `gpt-3` and `eval-harness` patterns. Moreover, on all datasets and for all patterns, models are sensitive to the number of demonstrations and do not necessarily improve with more demonstrations.



(a) 2 samples – `minimal` (b) 2 samples – `gpt-3` (c) 2 samples – `eval-harness`

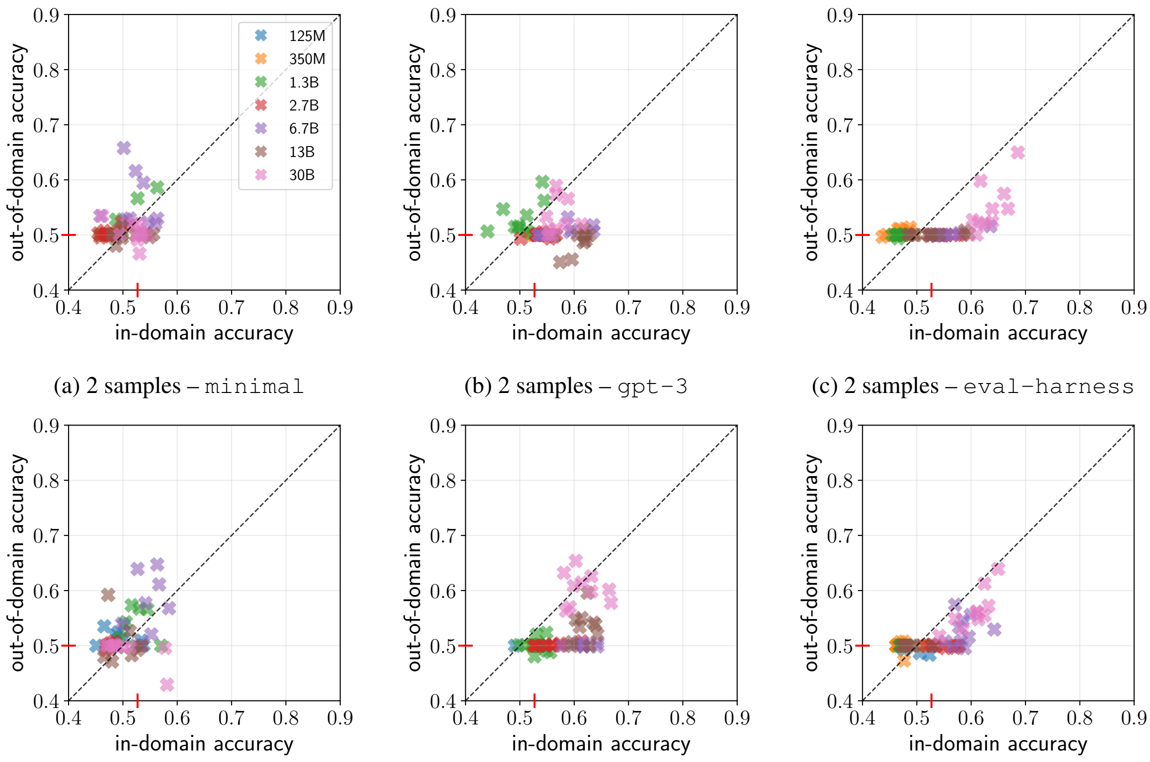
(d) 16 samples – `minimal` (e) 16 samples – `gpt-3` (f) 16 samples – `eval-harness`

(g) 32 samples – `minimal` (h) 32 samples – `gpt-3` (i) 32 samples – `eval-harness`

Figure 8: **Relationship between in-domain and out-of-domain performance of ICL on MNLI** for OPT models of various sizes. Rows vary amount of training data. Columns vary input pattern. Colors indicate model size. We run 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.

## B.3 Fine-tuning

We provide all FT results in Figures 11, 12, and 13. When comparing results across rows, we see the impact of the number of training examples on the results. Comparing results across columns demonstrates



(a) 2 samples – `minimal` (b) 2 samples – `gpt-3` (c) 2 samples – `eval-harness`

(d) 16 samples – `minimal` (e) 16 samples – `gpt-3` (f) 16 samples – `eval-harness`

Figure 9: **Relationship between in-domain and out-of-domain performance of ICL on RTE** for OPT models of various sizes. Rows vary amount of training data. Columns vary input pattern. Colors indicate model size. We run 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.

the importance of model selection for in-domain and out-of-domain performance.

Figures 14 and 15 show a comparison between two different ways of binarizing MNLI. For our main experiments, we remove the neutral class entirely. Merging it with the contradiction class instead leads to an even better relationship between in-domain and OOD performance.

## C Additional results for Pythia models

Figure 16 compares FT and ICL of Pythia models ranging from 410M to 12B parameters (Biderman et al., 2023). Similar to OPT, the Pythia models differ only in their size and have all been trained on exactly the same data (even in the exact same order). We focus on RTE and report results using 16 examples. For ICL, we use three different patterns (`minimal`, `gpt-3`, `eval-harness`). For FT, we report results using 16 and 128 examples and three different model selection strategies (best in-domain, last checkpoint, best out-of-domain). Significance tests are provided in Tables 2 and 11 to 13.

For ICL, all models perform poorly when using the `minimal` pattern. With the `gpt-3` pattern, we can observe a clear impact of model size on in-domain and out-of-domain performance. On the other hand, with the `eval-harness` pattern, for Pythia models, only in-domain performance improves with model size.

For FT, when using 16 samples and selecting checkpoints according to out-of-domain performance, almost all checkpoints lead to better out-of-domain than in-domain performance. Moreover, almost all fine-tuned models perform significantly better OOD than models adapted via ICL. When fine-tuning with 128 examples, we can see a very clear effect of model size on both in-domain and out-of-domain performance. In particular, when selecting checkpoints according to out-of-domain performance, almost all models perform better out-of-domain than in-domain.

|  |  | **FT** |  |  |  |  |
|---|---|---|---|---|---|---|
|  |  | **410M** | **1.4B** | **2.8B** | **6.9B** | **12B** |
| **ICL** | **410M** | 0.05 | 0.06 | 0.06 | 0.09 | 0.07 |
|  | **1.4B** | 0.03 | 0.04 | 0.04 | 0.07 | 0.05 |
|  | **2.8B** | −0.02 | −0.00 | −0.01 | 0.02 | 0.01 |
|  | **6.9B** | −0.03 | −0.02 | −0.02 | 0.01 | −0.01 |
|  | **12B** | −0.04 | −0.03 | −0.03 | −0.00 | −0.02 |

Table 11: Difference between average **in-domain performance** of ICL and FT with Pythia models on RTE. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to in-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.

|  |  | **FT** |  |  |  |  |
|---|---|---|---|---|---|---|
|  |  | **410M** | **1.4B** | **2.8B** | **6.9B** | **12B** |
| **ICL** | **410M** | −0.00 | 0.04 | 0.02 | 0.06 | 0.06 |
|  | **1.4B** | −0.02 | 0.02 | 0.00 | 0.04 | 0.04 |
|  | **2.8B** | −0.06 | −0.03 | −0.04 | −0.01 | −0.01 |
|  | **6.9B** | −0.08 | −0.04 | −0.06 | −0.02 | −0.02 |
|  | **12B** | −0.09 | −0.05 | −0.07 | −0.03 | −0.03 |

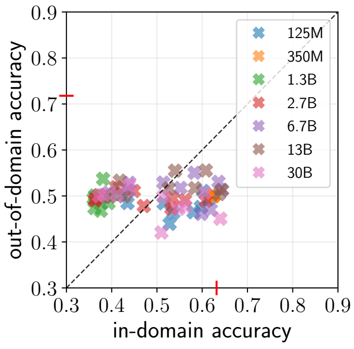
Table 12: Difference between average **in-domain performance** of ICL and FT with Pythia models on RTE. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to out-of-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.

|  |  | **FT** |  |  |  |  |
|---|---|---|---|---|---|---|
|  |  | **410M** | **1.4B** | **2.8B** | **6.9B** | **12B** |
| **ICL** | **410M** | 0.05 | 0.08 | 0.13 | 0.15 | 0.14 |
|  | **1.4B** | 0.04 | 0.07 | 0.12 | 0.14 | 0.13 |
|  | **2.8B** | −0.00 | 0.03 | 0.08 | 0.10 | 0.09 |
|  | **6.9B** | 0.04 | 0.07 | 0.12 | 0.14 | 0.13 |
|  | **12B** | 0.00 | 0.03 | 0.08 | 0.10 | 0.09 |

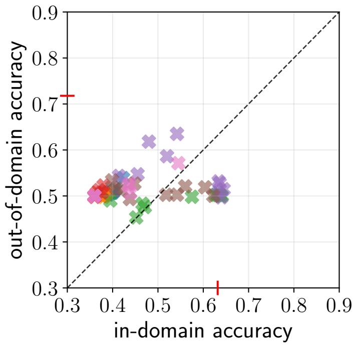
Table 13: Difference between average **out-of-domain performance** of ICL and FT with Pythia models on RTE. We use 16 examples and 10 random seeds for both approaches. For ICL, we use the `gpt-3` pattern. For FT, we use pattern-based fine-tuning (PBFT) and select checkpoints according to out-of-domain performance. We perform a Welch's t-test and color cells according to whether: ICL performs significantly better than FT, FT performs significantly better than ICL. For cells without color, there is no significant difference between ICL and FT.

## D Analyzing individual OPT fine-tuning runs

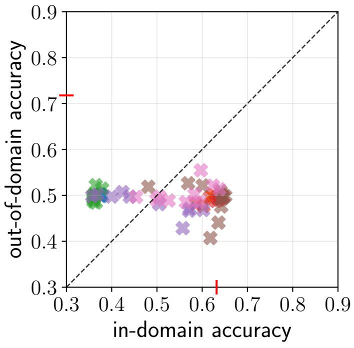
Looking at the in-domain and out-of-domain performance for individual checkpoints does not reveal the generalization behavior of individual FT runs during training. In particular, this view does not tell us how stable the generalization of individual runs is during FT. Therefore, in Figures 17 and 18 we visualize both in-domain and out-of-domain performance throughout FT on MNLI and RTE when using 128 examples. We observe that out-of-domain performance varies considerably across seeds and even during fine-tuning.



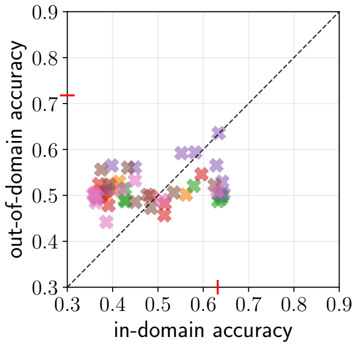
(a) 2 samples – `minimal`



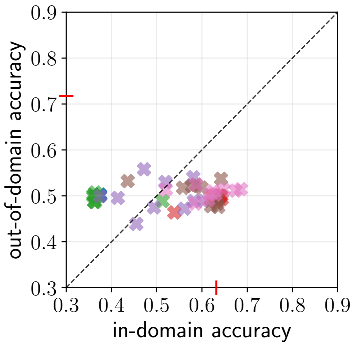
(b) 2 samples – `eval-harness`



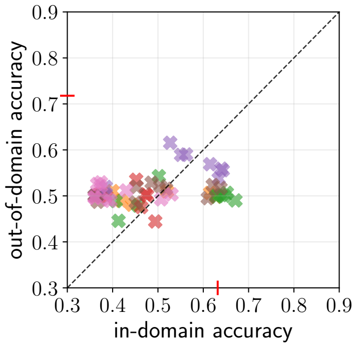
(c) 16 samples – `minimal`



(d) 16 samples – `eval-harness`

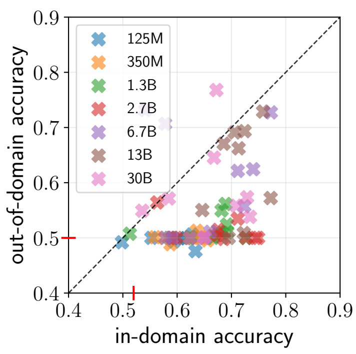


(e) 32 samples – `minimal`

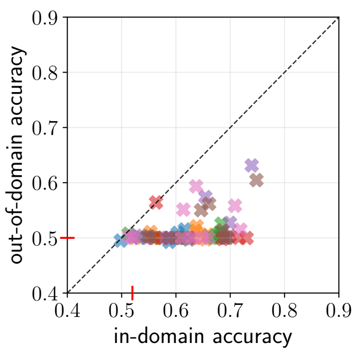


(f) 32 samples – `eval-harness`

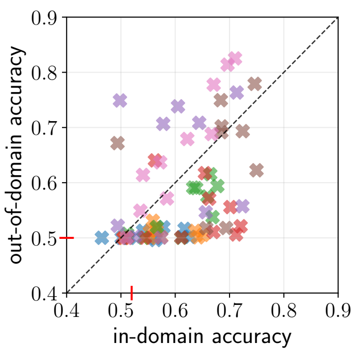
Figure 10: **Relationship between in-domain and out-of-domain performance of ICL on QQP** for OPT models of various sizes. Rows vary amount of training data. Columns vary input pattern. Colors indicate model size. We run 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.



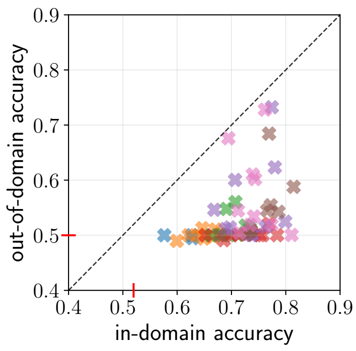
(a) 16 – in-domain



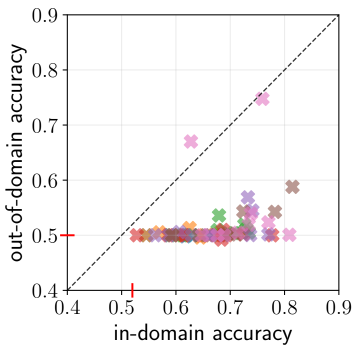
(b) 16 – last checkpoint



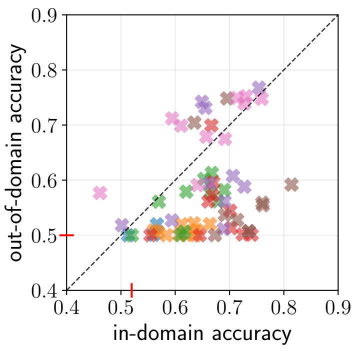
(c) 16 – out-of-domain



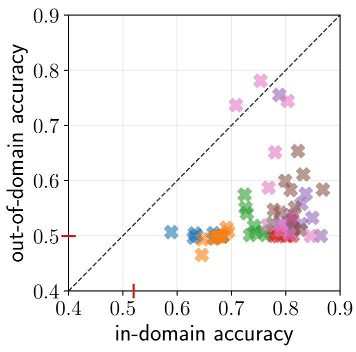
(d) 32 – in-domain



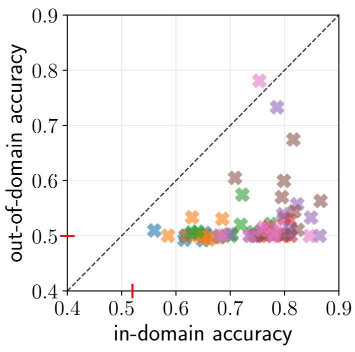
(e) 32 – last checkpoint



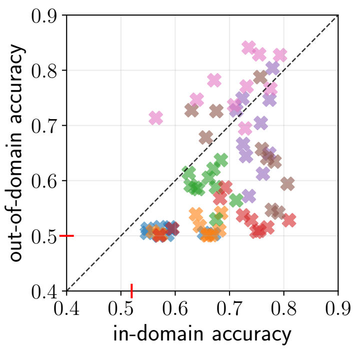
(f) 32 – out-of-domain



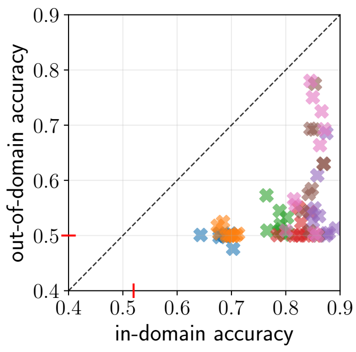
(g) 64 – in-domain



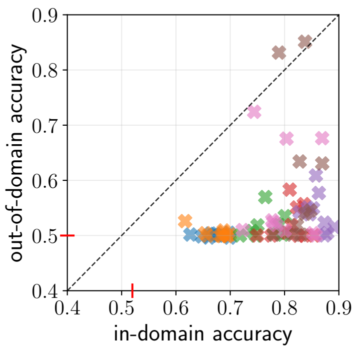
(h) 64 – last checkpoint



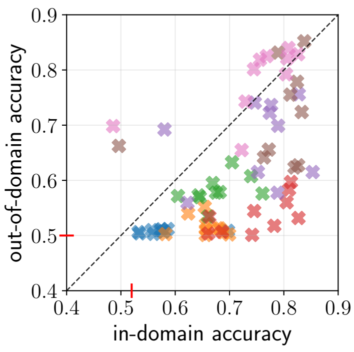
(i) 64 – out-of-domain



(j) 128 – in-domain

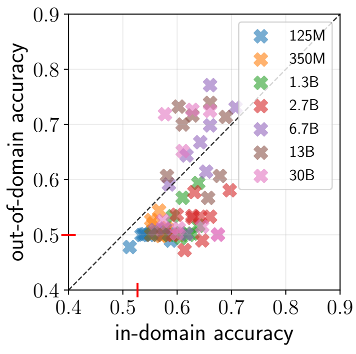


(k) 128 – last checkpoint

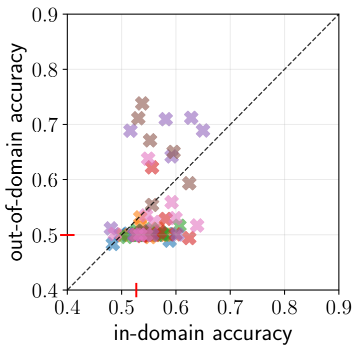


(l) 128 – out-of-domain

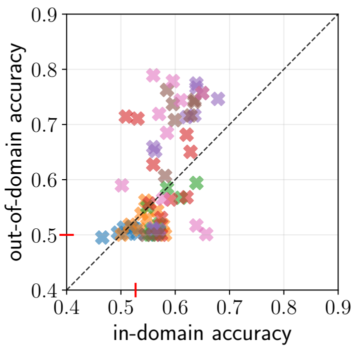
Figure 11: **Relationship between in-domain and out-of-domain performance of PBFT on MNLI** for OPT models of various sizes. Rows vary amount of training data. Columns vary model selection strategy. Colors indicate model size. We fine-tune 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.



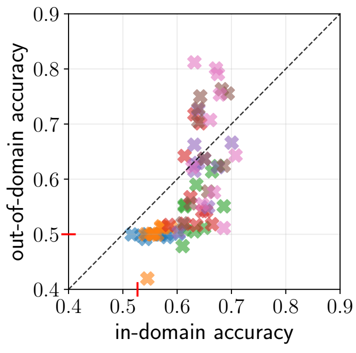
(a) 16 – in-domain



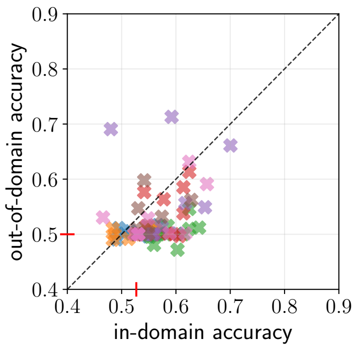
(b) 16 – last checkpoint



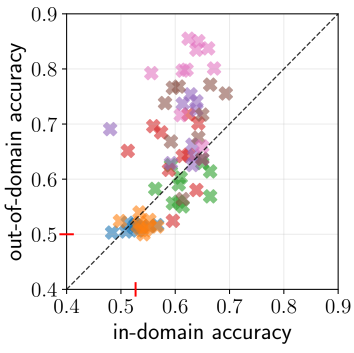
(c) 16 – out-of-domain



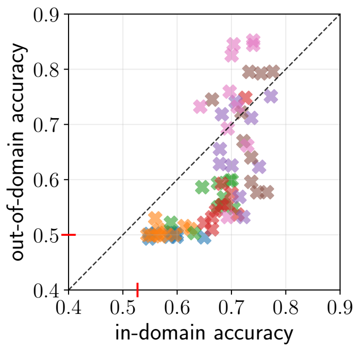
(d) 32 – in-domain



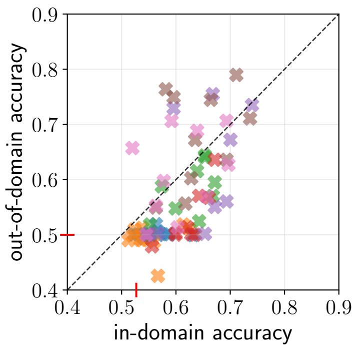
(e) 32 – last checkpoint



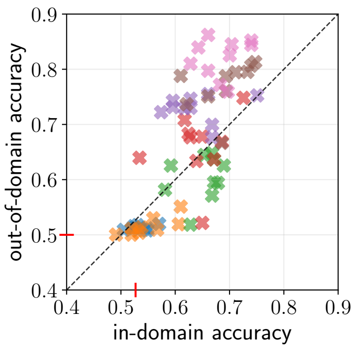
(f) 32 – out-of-domain



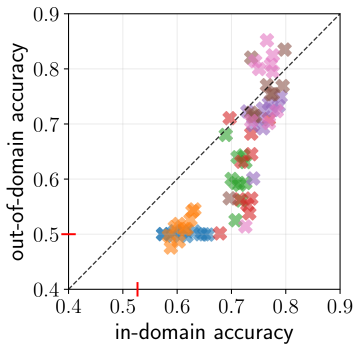
(g) 64 – in-domain



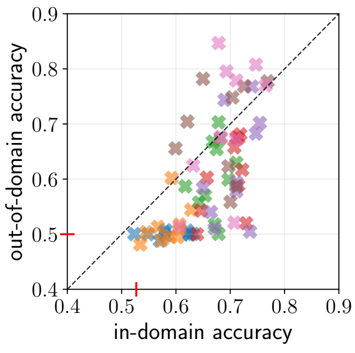
(h) 64 – last checkpoint



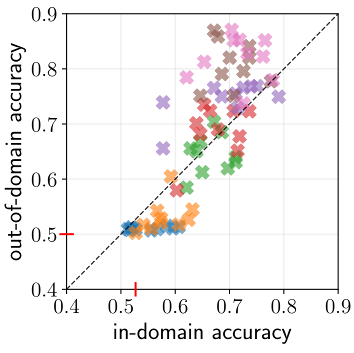
(i) 64 – out-of-domain



(j) 128 – in-domain

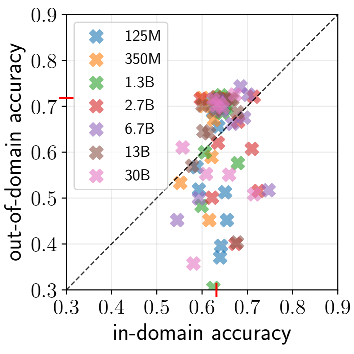


(k) 128 – last checkpoint

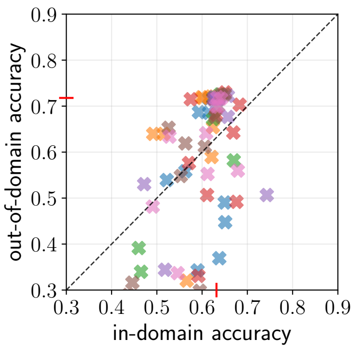


(l) 128 – out-of-domain

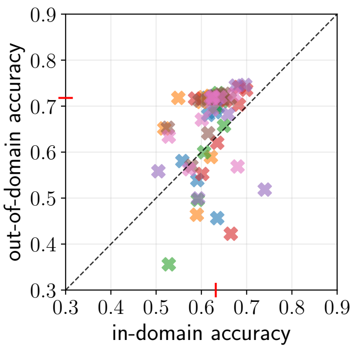
Figure 12: **Relationship between in-domain and out-of-domain performance of PBFT on RTE** for OPT models of various sizes. Rows vary amount of training data. Columns vary model selection strategy. Colors indicate model size. We fine-tune 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.



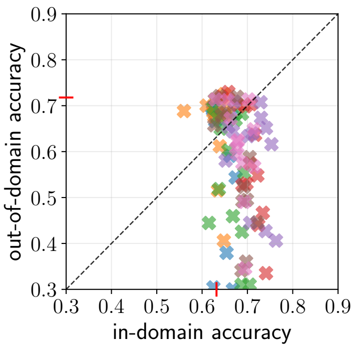
(a) 16 – in-domain



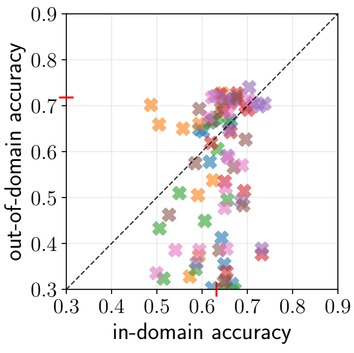
(b) 16 – last checkpoint



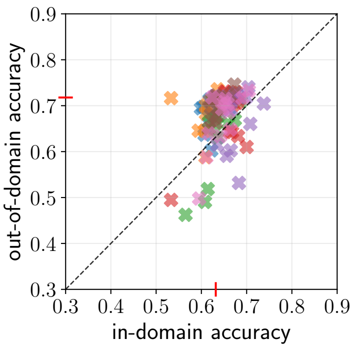
(c) 16 – out-of-domain



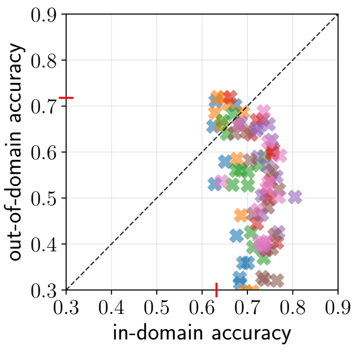
(d) 32 – in-domain



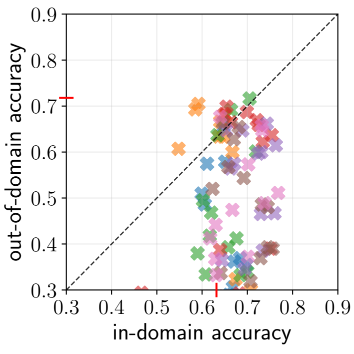
(e) 32 – last checkpoint



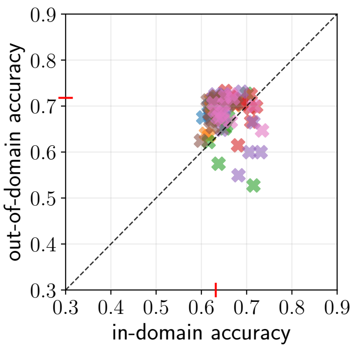
(f) 32 – out-of-domain



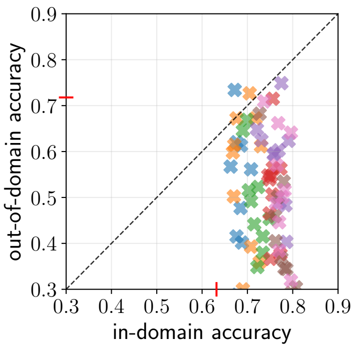
(g) 64 – in-domain



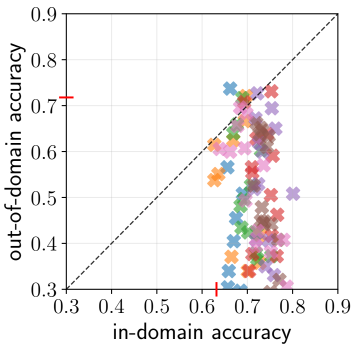
(h) 64 – last checkpoint



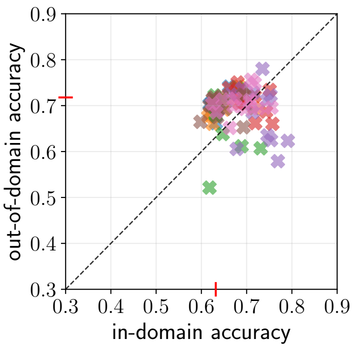
(i) 64 – out-of-domain



(j) 128 – in-domain

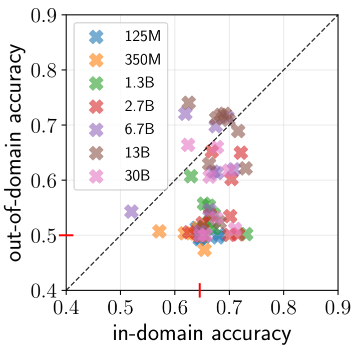


(k) 128 – last checkpoint

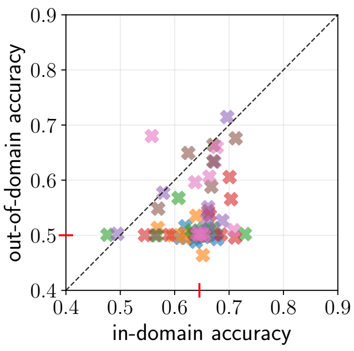


(l) 128 – out-of-domain

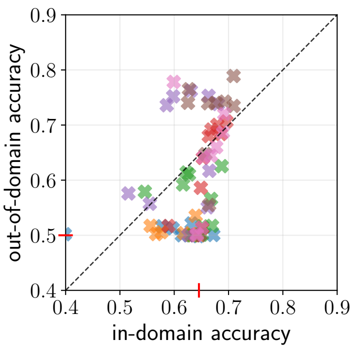
Figure 13: **Relationship between in-domain and out-of-domain performance of PBFT on QQP** for OPT models of various sizes. Rows vary amount of training data. Columns vary model selection strategy. Colors indicate model size. We fine-tune 10 models per setting varying only the data seed. — in the x- and y-axis indicates the performance of the majority class label.



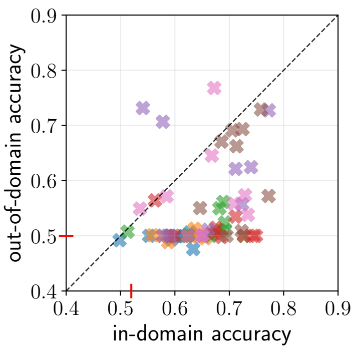
(a) *merge* – in-domain



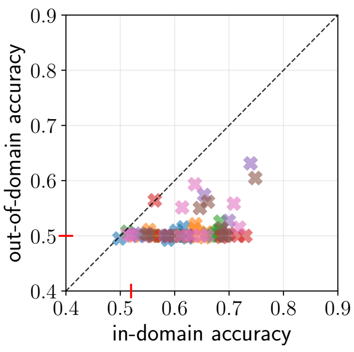
(b) *merge* – last



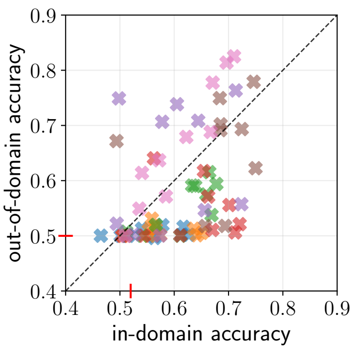
(c) *merge* – out-of-domain



(d) *remove* – in-domain

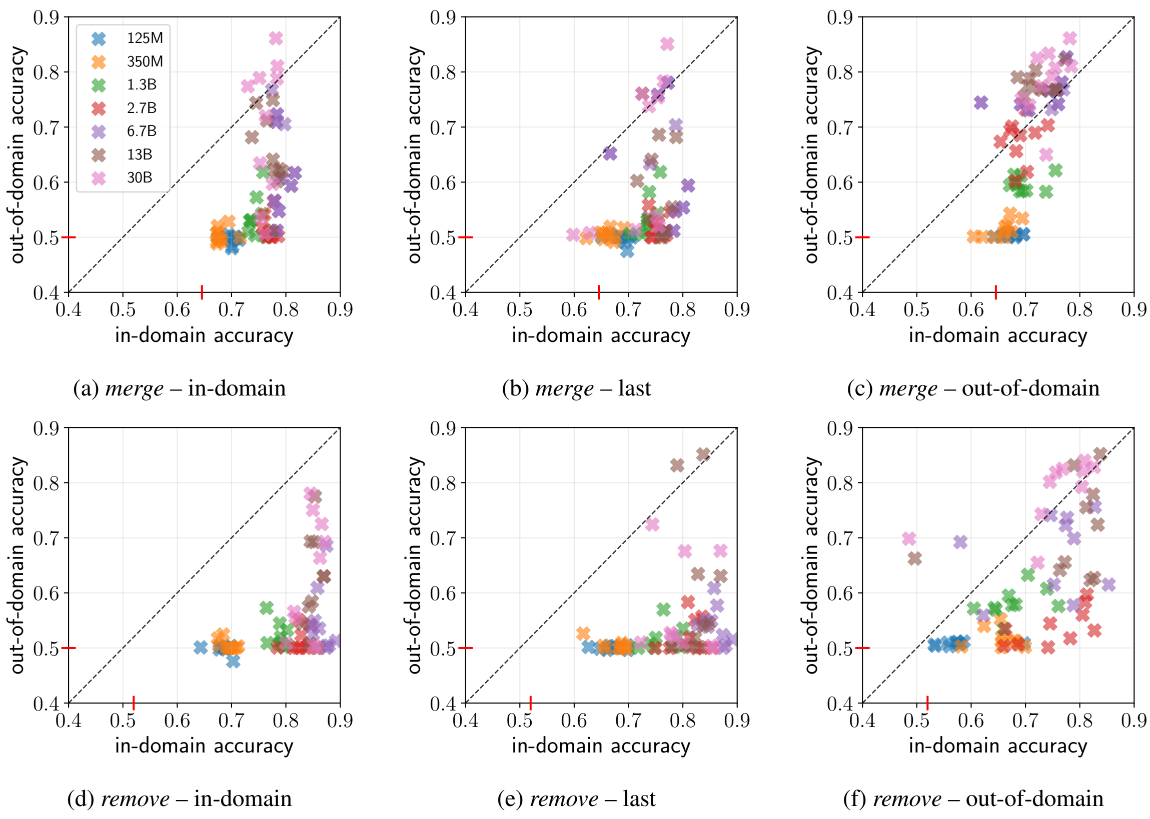


(e) *remove* – last



(f) *remove* – out-of-domain

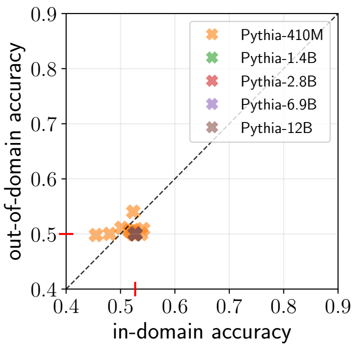
Figure 14: **Relationship between in-domain and out-of-domain performance of PBFT on MNLI** for OPT models of various sizes when **merging** the neutral and contradiction classes **vs. removing** the neutral examples altogether. We fine-tune on **16 examples** using 10 different seeds. — in the x- and y-axis indicates the performance of the majority class label.



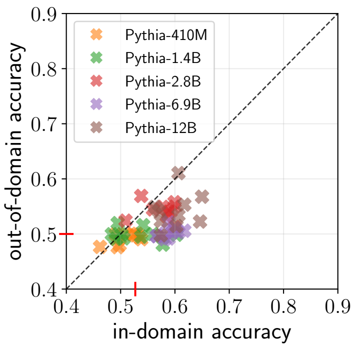
(a) *merge* – in-domain (b) *merge* – last (c) *merge* – out-of-domain

(d) *remove* – in-domain (e) *remove* – last (f) *remove* – out-of-domain

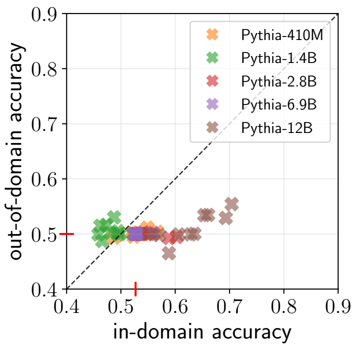
Figure 15: **Relationship between in-domain and out-of-domain performance of PBFT on MNLI** for OPT models of various sizes when **merging** the neutral and contradiction classes **vs. removing** the neutral examples altogether. We fine-tune on **128 examples** using 10 different seeds. — in the x- and y-axis indicates the performance of the majority class label.



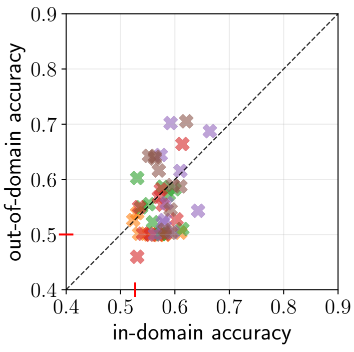
(a) ICL 16 samples – `minimal`



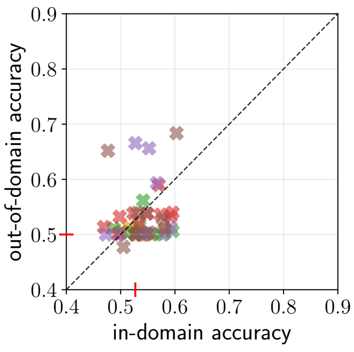
(b) ICL 16 samples – `gpt-3`



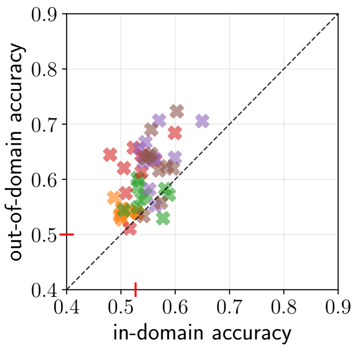
(c) ICL 16 samples – `eval-harness`



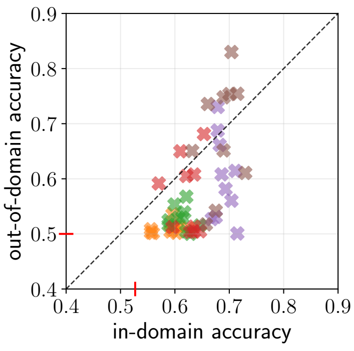
(d) FT 16 samples – best in-domain



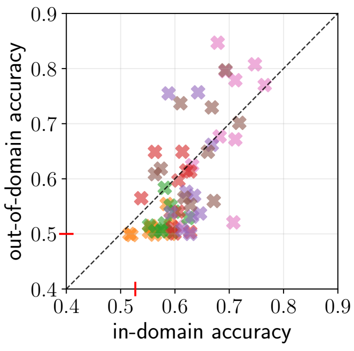
(e) FT 16 samples – last



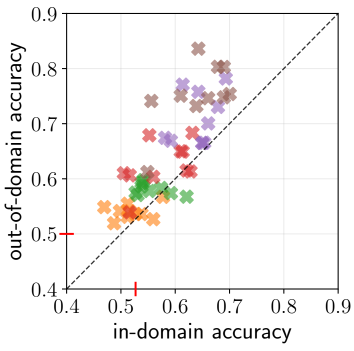
(f) FT 16 samples – best out-of-domain



(g) FT 128 samples – best in-domain

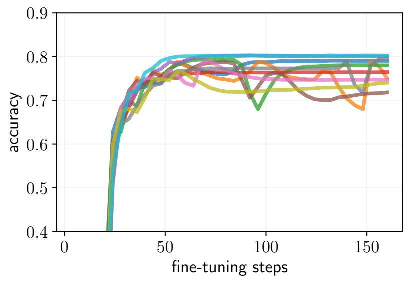


(h) FT 128 samples – last

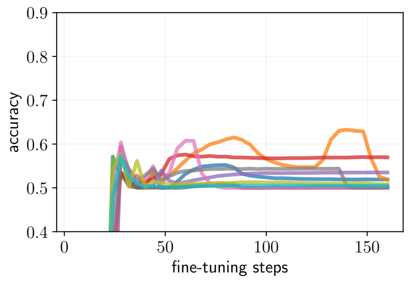


(i) FT 128 samples – best out-of-domain

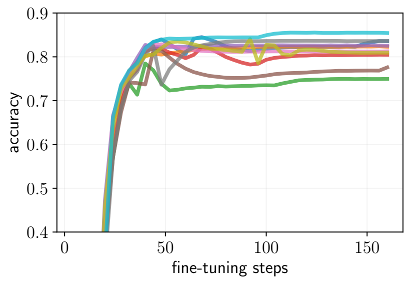
Figure 16: **ICL and FT results for Pythia models of different size**. For ICL, we report results using 16 examples and three different patterns (`minimal`, `gpt-3`, `eval-harness`). For FT, we report results using 16 and 128 examples using three different model selection strategies (best in-domain, last checkpoint, best out-of-domain). In all cases, we show results for 10 different random seeds. — in the x- and y-axis indicates the performance of the majority class label.



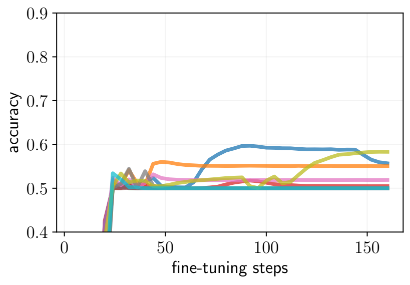
(a) 1.3B – in-domain



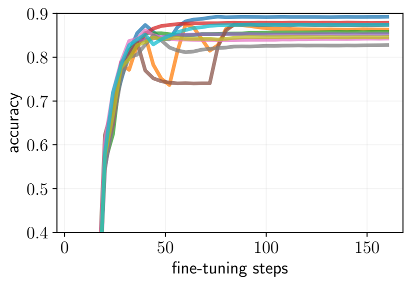
(b) 1.3B – out-of-domain



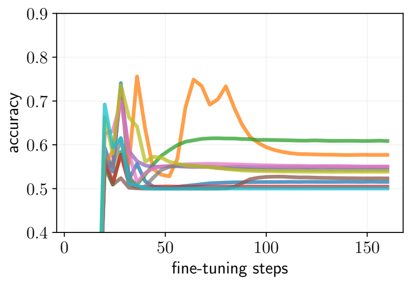
(c) 2.7B – in-domain



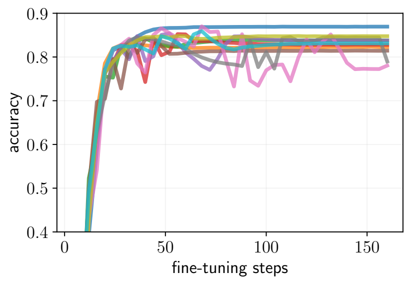
(d) 2.7B – out-of-domain



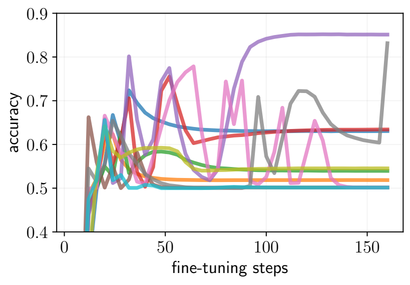
(e) 6.7B – in-domain



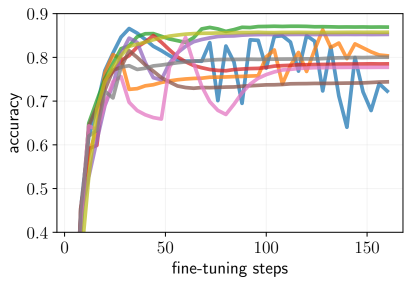
(f) 6.7B – out-of-domain



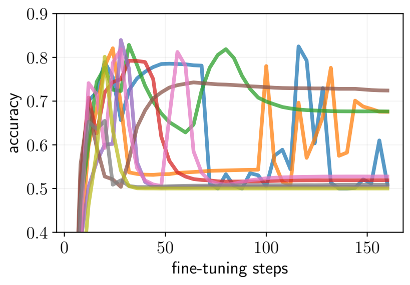
(g) 13B – in-domain



(h) 13B – out-of-domain

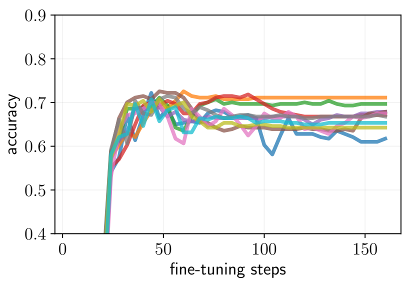


(i) 30B – in-domain

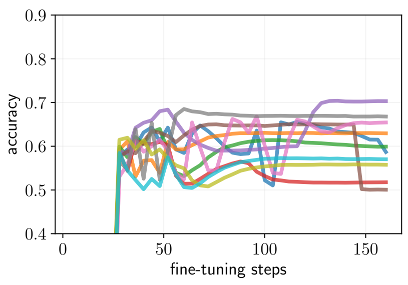


(j) 30B – out-of-domain

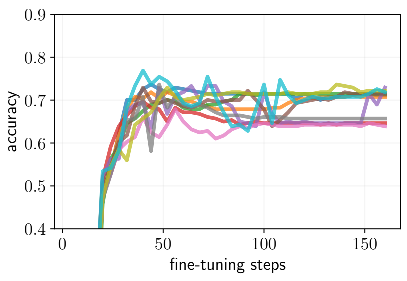
Figure 17: **Generalization throughout PBFT on MNLI for OPT models of various sizes.** We train on 128 examples. Colors denote different data seeds. First column shows in-domain, second column out-of-domain performance.



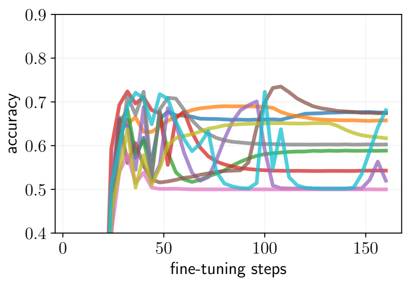
(a) 1.3B – in-domain



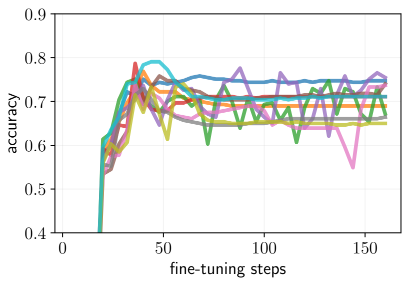
(b) 1.3B – out-of-domain



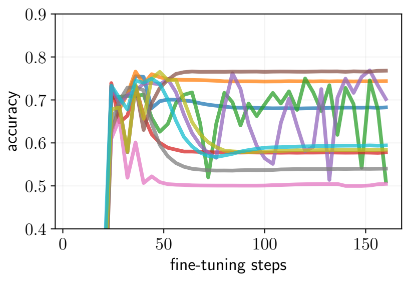
(c) 2.7B – in-domain



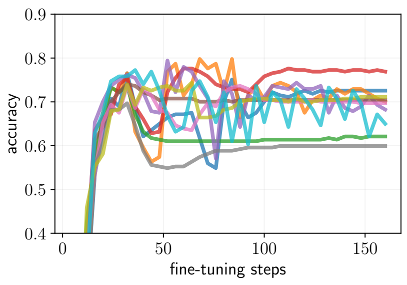
(d) 2.7B – out-of-domain



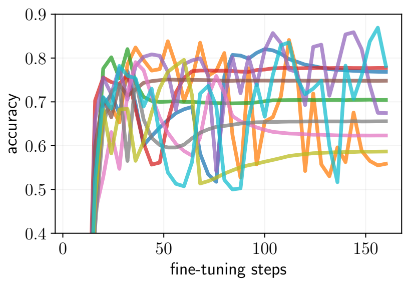
(e) 6.7B – in-domain



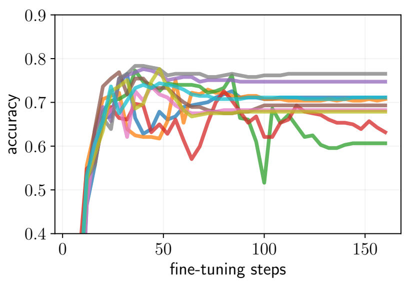
(f) 6.7B – out-of-domain



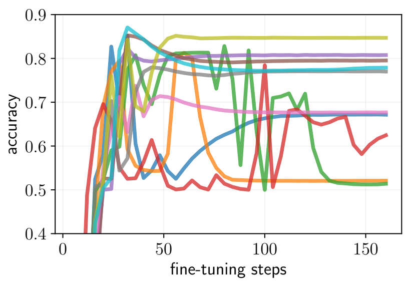
(g) 13B – in-domain



(h) 13B – out-of-domain



(i) 30B – in-domain



(j) 30B – out-of-domain

Figure 18: **Generalization throughout PBFT on RTE for OPT models of various sizes.** We train on 128 examples. Colors denote different data seeds. First column shows in-domain, second column out-of-domain performance.

## ACL 2023 Responsible NLP Checklist

### A For every submission:

☑ A1. Did you describe the limitations of your work?
*Section 9*

☒ A2. Did you discuss any potential risks of your work?
*Our work is an analysis and fair comparison of existing methods.*

☑ A3. Do the abstract and introduction summarize the paper's main claims?
*Abstract and Introduction (Section 1)*

☒ A4. Have you used AI writing assistants when working on this paper?
*Left blank.*

### B ☐ Did you use or create scientific artifacts?

*Not applicable. Left blank.*

☑ B1. Did you cite the creators of artifacts you used?
*Section 2 and 3*

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*Not applicable. Left blank.*

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
*Not applicable. Left blank.*

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
*Not applicable. Left blank.*

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
*Not applicable. Left blank.*

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
*Left blank.*

### C ☑ Did you run computational experiments?

*Section 4*

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
*Section 3 and Appendix*

*The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.*

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
*Section 3 and Appendix*

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
*Section 4 and Appendix*

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
*Not applicable. Left blank.*

**D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?**

*Left blank.*

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
*Not applicable. Left blank.*

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
*Not applicable. Left blank.*

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
*Not applicable. Left blank.*

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
*Not applicable. Left blank.*

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
*Not applicable. Left blank.*