Monte Carlo Sampling for Analyzing In-Context Examples

Stephanie Schoch and **Yangfeng Ji**
Department of Computer Science
University of Virginia
Charlottesville, VA 22904
{sns2gr,yangfeng}@virginia.edu

Abstract

Prior works have shown that in-context learning is brittle to presentation factors such as the order, number, and choice of selected examples. However, ablation-based guidance on selecting the number of examples may ignore the interplay between different presentation factors. In this work we develop a Monte Carlo sampling-based method to study the impact of number of examples while explicitly accounting for effects from order and selected examples. We find that previous guidance on how many in-context examples to select does not always generalize across different sets of selected examples and orderings, and whether one-shot settings outperform zero-shot settings is highly dependent on the selected example. Additionally, inspired by data valuation, we apply our sampling method to in-context example selection to select examples that perform well across different orderings. We find a negative result, that while performance is robust to ordering and number of examples, there is an unexpected performance degradation compared to random sampling.

1 Introduction

In-context learning is an emergent ability of LLMs (Wei et al., 2022; Brown et al., 2020b) where an LLM learns to perform an unseen task by seeing a number of demonstrations in the context window (Brown et al., 2020a). While in-context learning has shown significant potential as a way to extract relevant information from an LLM and align the model with user expectations, it has also exhibited brittleness to an assortment of factors. For example, model performance when learning in-context is sensitive to which examples are selected (Rubin et al., 2022; Liu et al., 2022; Wu et al., 2023; Ye et al., 2023) as well as their orderings (Lu et al., 2022; Chen et al., 2023b; Liu et al., 2022; Chang and Jia, 2023; Guo et al., 2024; Wu et al., 2023).

Another important parameter, the number of examples, has received comparably little attention. Prior works have suggested that one-shot settings outperform zero-shot settings even when a random label is used (Min et al., 2022). Additional ablations have guided this parameter by citing performance plateaus at set numbers of examples (Wang et al., 2024; Min et al., 2022; Wu et al., 2023). However, it is unclear whether this guidance holds when accounting for other sensitive factors such as different orderings and selected examples.

Previous work on data valuation has shown the efficacy of applying Monte Carlo sampling to evaluate datum contributions under different permutations in fine-tuning settings (Ghorbani and Zou, 2019; Schoch et al., 2023). Inspired by this, we develop a Monte Carlo sampling-based method to investigate the impact of number of examples while using permutations to account for order and selection of in-context examples.

Specifically, we utilize Monte Carlo sampling and analyze performance with the addition of each exemplar. We find that performance plateaus at previously suggested numbers of examples do not consistently generalize under different permutations. Further, we find that one-shot performance may be more sensitive to the selected example than previously recognized (Min et al., 2022; Brown et al., 2020b) and the guidance of one-shot outperforming zero-shot is dependent on the selected example. Finally, we find that using Monte Carlo sampling to select in-context examples may increase robustness to effects from ordering and selected examples, but unexpectedly, does not lead to performance improvements over random sampling.

2 Related work

In-context learning In many prior works investigating in-context learning sensitivity to ordering and selected examples (see Appendix A for de-

scription of in-context learning), a fixed number of examples are used (Zhang et al., 2022b; Lu et al., 2022; Min et al., 2022), with common guidance from prior ablations stating performance plateaus around $k = 4$ (Wang et al., 2024) and $k = 8$ examples (Min et al., 2022; Wu et al., 2023). Recent work has looked at the how the number of examples impacts performance on chain-of-thought reasoning benchmarks (Chen et al., 2023a) and suggested that fewer examples may be needed, yet it is otherwise unclear the effect of number of demonstrations on other tasks and prompting frameworks, particularly when controlling for order and selected examples.

Monte Carlo sampling Monte Carlo sampling has been widely adopted in the data valuation literature to provide unbiased approximations of the Shapley value (Ghorbani and Zou, 2019), based on prior work on Monte Carlo methods for Shapley value approximation (Mann and Shapley, 1962; Castro et al., 2009; Maleki et al., 2013) (see Appendix A for additional details). Recent works have also applied principles from data valuation to in-context learning example selection and ordering (Guo et al., 2024; Chang and Jia, 2023; Nguyen and Wong, 2023). While in data valuation, Monte Carlo sampling methods are used to calculate the marginal contribution of each data point averaged over a number of permutations, our motivation differs. In our setting, we are motivated by the utility of Monte Carlo sampling to provide an unbiased estimate of the influence of number of examples on in-context learning performance by reducing influence of ordering and selected examples.

3 Method

To study the effect of the number of examples, we aim to reduce the influence of ordering and selected examples as confounding factors. While averaging across trials in previous work helps reduce the influence of selected examples, it does not account for different orderings. Additionally, prior work on ordering only addresses up to $k = 4$ (Lu et al., 2022) as the possible permutations increase exponentially with respect to k. This motivates the use of Monte Carlo sampling with incremental exemplar additions as 1) the use of permutations reduces the influence of ordering, and 2) the averaging across multiple trials reduces the influence of selected examples. We explain this as well as our method in detail below.

Algorithm 1: Monte Carlo Sampling Method

Input: Training data $D_{trn} = \{1, ..., n\}$, evaluation data $D_{tst} = \{1, ..., m\}$, LLM $\mathcal{M}$, performance metric $V_{\mathcal{M}}$, parameters: K (# examples), P (# permutations)
Output: Average performance $\mu(V_{\mathcal{M}})$ for $k = \{0, ..., K\}$ for one subset S^K
Initialize $\mu_k = 0$ for $k = 0, ..., K$
Randomly sample subset S^K from D_{trn}
for $p \in \{1, ..., P\}$ **do**
  π^p: Random permutation of S^K
  for $k \in \{0, ..., K\}$ **do**
    $\mu_k \leftarrow \mu_k + V_{\mathcal{M}}(S[0:k]_\pi)$
  end for
end for
for $k \in \{0, ..., K\}$ **do**
  $\mu_k \leftarrow \frac{1}{P}\mu_k$
end for

Consider a training dataset $D_{trn} = \{(x_i, y_i)\}_{i=1}^n$ that contains n training instances. Given a fixed test set D_{tst}, we aim to draw k exemplars from D_{trn} and test model performance on D_{tst}. We let $S_\pi^k \subseteq D_{trn}$ denote a subset with some ordering π and cardinality k, where $k \leq n$. Additionally, we let $V_{\mathcal{M}}$ denote the predictive performance of a model $\mathcal{M}$, e.g., prediction accuracy.

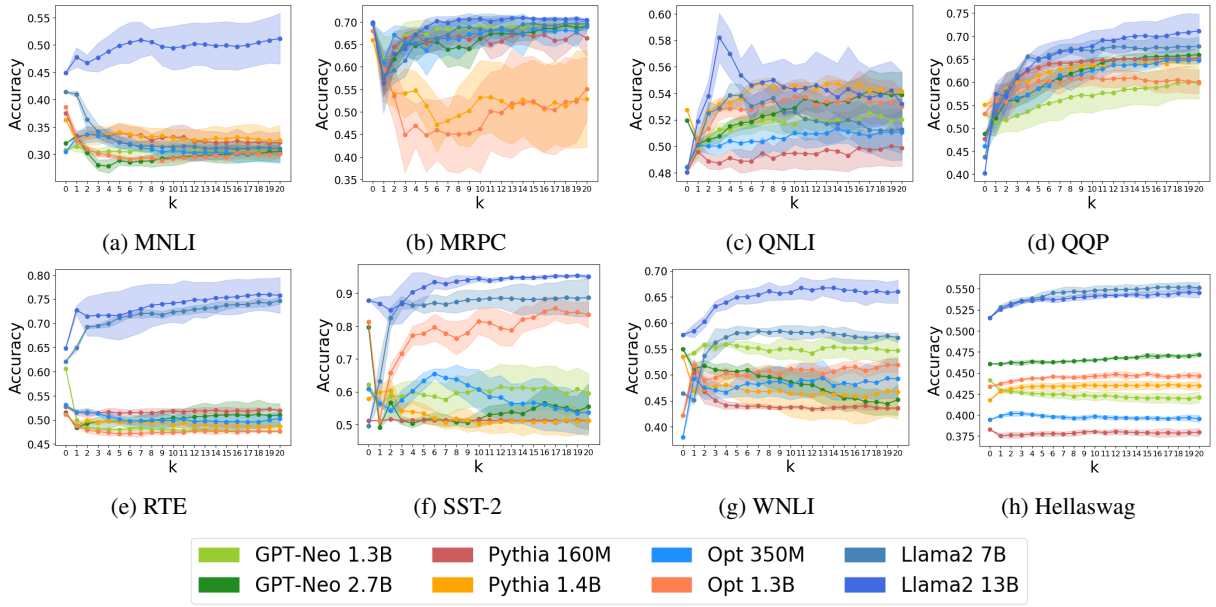
For each $k \in \{1, 2, ..., K\}$, the set of possible subsets $S^k = \{S_{(i)}^k\}_{i=1}^{{}_nC_k}$. The expected model performance for any given $S_{(i)}^k$ can be defined as:

$$E_{\pi \sim \Pi}[V_{\mathcal{M}}(S_{\pi(i)}^k)] \quad (1)$$

where $V_{\mathcal{M}}(S_\pi^k)$ is the performance of model $\mathcal{M}$ using S_π^k and Π is the uniform distribution over all $k!$ permutations of $S_{(i)}^k$.

For each k, exhaustively permuting all ${}_nC_k$ subsets S^k, and averaging over the expected values would give us the expected value for each number of exemplars. In practice, however, this is computationally infeasible. Instead, we utilize Monte Carlo sampling to approximate this value.

Specifically, we utilize the Monte Carlo sampling method adapted from Ghorbani and Zou (2019). First, we define K as the maximum exemplars we aim to use. Next, we sample S^K from D_{trn}. For a predefined number of permutations of S^K, we iteratively scan from the first element to the final element, computing the model perfor-



(a) MNLI (b) MRPC (c) QNLI (d) QQP

(e) RTE (f) SST-2 (g) WNLI (h) Hellaswag

Figure 1: In-context performance for each dataset and model. Results show the average of 20 permutations at each step k in the proposed Monte Carlo sampling method. Shaded regions show standard deviation of 5 trials.

mance at each step. We average the performance for each k across p permutations. We provide the pseudo-code in algorithm 1. In practice, we perform this procedure over multiple trials, resampling S^K for each trial and averaging over the result. In our experiments, we use 5 trials and $p = 20$ permutations.

In addition to the computational speedup achievable by limiting the number of permutations, the primary benefit of our approach is that is circumvents the need to resample at each k. As we cannot exhaustively permute each S^k, if we resample for each k, there are $\frac{n!}{(n-k-1)!}$ possible permutations preceding k, each of which could produce differing downstream performance $V_{\mathcal{M}}(S^{k-1}_{\pi_{k-1}})$ due to the impact of S^{k-1} and π_{k-1} (selected examples and ordering, respectively). By sampling in this manner, we effectively eliminate the influence of $k-1$ in understanding performance at k.

4 Experiment setup

Detailed descriptions of our setup can be found in Appendix B.

Models We experiment with 8 models in total, listed in Table 1. The selected models vary in size from 160M to 13B and represent four distinct families: Pythia (Biderman et al., 2023), OPT (Zhang et al., 2022a), GPT-Neo (Black et al., 2021), and Llama2 (Touvron et al., 2023).

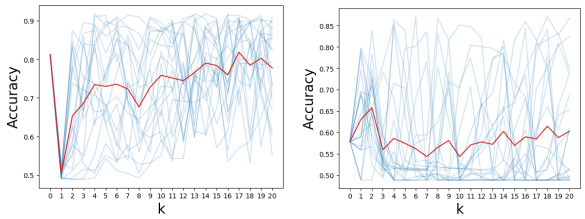
Datasets We use 8 datasets in our experiments across a diverse range of tasks previously represented in in-context learning analysis. Specifically, we perform experiments on natural language inference (MNLI, QNLI, WNLI), sentiment analysis (SST-2), commonsense reasoning (Hellaswag), paraphrase detection (MRPC, QQP), and textual entailment (RTE).

Sampling and Aggregation Procedure For each trial, we randomly sample 20 in-context examples from the training set. We perform 20 Monte Carlo permutations. Within each permutation, we iterate from the first example to the last, computing the accuracy on the validation set at each step. Using this procedure, we perform 5 trials for each experiment. In aggregate, we perform 100 permutations with each model and dataset combination.

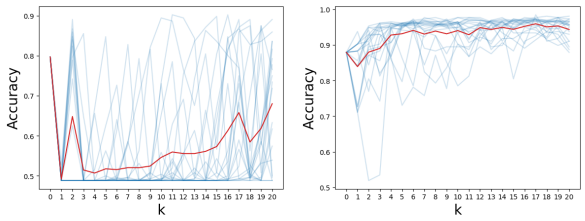
5 Results

5.1 Analyzing Number of Examples

We plot the performance of each model across $k = \{1, 2, ..., 20\}$ examples, where each step represents the addition of one in-context example from the Monte Carlo permutation. In line with common practice of reporting across multiple trials with standard deviation information, our results are averaged over 5 trials, controlling for selected examples and ordering effects. Results are reported in Figure 1.

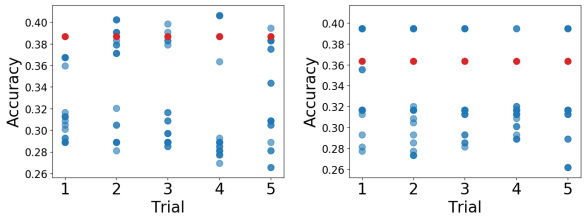


(a) Opt-1.3B (b) Pythia-1.4B

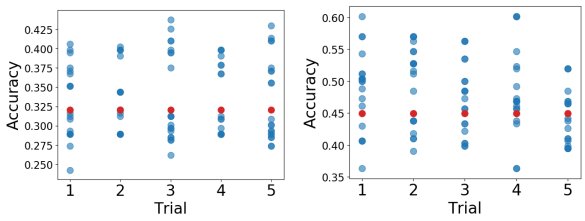


(c) GPT-Neo-2.7B (d) Llama2-13B

Figure 2: Results on SST-2. Blue lines represent individual permutations and red line indicates average across all permutations within one trial.



(a) Opt-1.3B (b) Pythia-1.4B



(c) GPT-Neo-2.7B (d) Llama2-13B

Figure 3: One-shot MNLI performance across 5 trials. Each blue point represents the accuracy using the first exemplar in a permutation. Red points indicate zero-shot performance. Results show that zero-shot settings can outperform one-shot settings, dependent upon the selected example.

Observed from the averaged results, our results align with ablations in prior work (Wu et al., 2023; Min et al., 2022). Specifically, much of the performance improvement across models and datasets occurs within the first 8 in-context examples, where performance then begins to plateau and only marginally increase. At face value, this result shows that prior recommendations with respect to the number of examples to use are not influenced by ordering effects as we had hypothesized prior to designing our sampling method.

Does performance consistently plateau at set numbers of exemplars? When viewing the results from individual permutations, we find that the performance plateaus by $k = 4$ (Wang et al., 2024) and $k = 8$ (Min et al., 2022; Wu et al., 2023) examples are not observable within individual permutations. On the contrary, we continue to observe erratic performance changes up through $k = 20$ examples. This suggests the previously observed plateaus on averaged results, both in prior work and recreated in Figure 1, may mask significant performance fluctuations, and the best performance within a selected example set may occur anywhere from $k = \{1, 2, ...20\}$. Additionally, we observe that in some permutations, there is a performance drop at $k = 1$. We investigate this further in the following section.

Is one example better than none? Prior work has suggested that one demonstration outperforms no demonstrations even when a random label is used (Min et al., 2022). Other work has suggested some dependence on the dataset and model (Xie et al., 2022; Brown et al., 2020b). However, our results indicate that the performance of zero-shot vs. few-shot settings may be dependent on the selected example, regardless of dataset and model. To illustrate this, in Figure 6, we plot the one-shot performance of each permutation across all 5 trials on the MNLI dataset using each model, with the zero-shot performance of the model indicated in red.

In contradiction to prior work, our results across nearly all models and datasets (see Appendix C) indicate that performance in one-shot settings can vary between significant performance improvements and significant performance degradation, depending on the selected example. This contradiction raises the question of what qualities of in-context examples can lead to such significant performance degradation in one-shot settings, and whether these have any impact when used within a k-shot setting.

Our analysis in one-shot settings across different permutations indicates that one-shot performance is more sensitive to the selected example than previously thought.

5.2 Exemplar Selection

We are interested in whether we can apply our sampling method to selecting in-context examples. Using our sampling method, each exemplar appears at

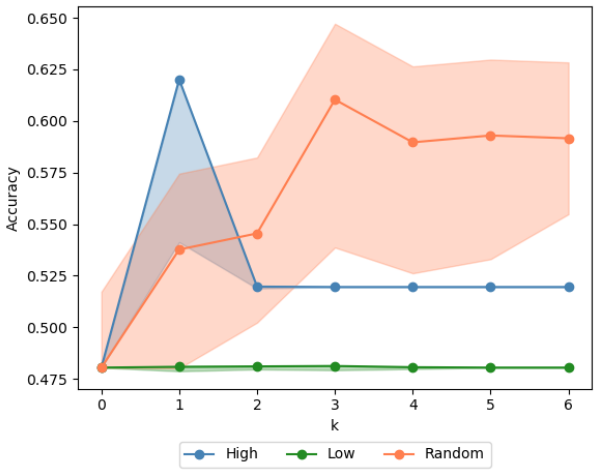


Figure 4: Performance with Llama2-13B on QNLI dataset, using in-context subsets containing the highest-performing and lowest-performing data points on average from subsection 5.1, along with a random baseline. Results represent 20 permutations, with standard deviation displayed as the shaded region for each line.

different k's under different permutations. Therefore, we are effectively averaging out different orderings and k's. It follows that we would expect exemplars associated with high average accuracy within different permutations to be associated with higher model performance overall, while being robust across orderings and different k's Further, we would expect exemplars with low average accuracies to consistently lead to poor performance.

Given the token limitations of context windows and computational time associated with increasing k, to increase the example candidate set, we use Z-score as a means to compare examples across trials. This allows us to limit context to $k = 20$ while increasing the overall candidate set and accounting for the impact of using different selected examples in each trial. We perform the sampling procedure described in section 3 with Llama2-13B on QNLI and perform the following calculation.

For a given trial t, we compute the average accuracy associated with each example e, denoted μ_e. We then take the mean over all example averages in the trial, $\mu_t = \frac{1}{k}\Sigma_{e=1}^{k}\mu_e$ with standard deviation $\sigma_t = \sqrt{\frac{\Sigma_{e=1}^{k}(\mu_e - \mu_t)}{k}}$. Finally, we compute the Z-score for each example in the trial, where $Z = \frac{\mu_e - \mu_t}{\sigma_t}$ After computing scores for all trials, we identify points with a Z-score $Z > 1 \vee Z < 1$, and select the 6 examples with the highest and lowest scores as our *high* and *low* set, respectively. We report results using each set, as well as a random baseline for in-context learning in Figure 4 using the same sampling procedure as in subsection 5.1.

When using points identified via Monte Carlo sampling as consistently high or low performing, we see a greater robustness to ordering sensitivity across k's, as evidenced by the minimal variance exhibited by these subsets. This is contrasted with the random baseline which exhibits high variance across different k's and orderings.

Interestingly, whereas our experiments in section 5.1 showed numerous instances where one-shot performance was worse than zero-shot performance, we do not observe this occurring with the lowest average performing data points. Rather, we see performance maintained at a zero-shot level. Further, while high-performing examples consistently performs above zero-shot accuracy and exhibit greater stability over varying k across permutations, the random selection exhibited the highest performance for $k > 1$. This was an unexpected result: using our Monte Carlo-based selection method led to more robust performance across k examples, but it resulted in an overall performance decrease compared to random sampling.

Our results when selecting in-context examples that on average had high or low scores across Monte Carlo permutations indicate that subsets comprised of points at each end of the spectrum exhibit lower sensitivity to ordering and number of demonstrations. While this may be promising in terms of identifying methods to increase robustness of in-context learning performance, the random selection baseline exhibited both higher overall performance and higher variance across orderings and number of examples. This raises the question of whether there is an existing performance vs. robustness tradeoff, and of what qualities contribute to example deviation from the mean performance, as well as how methods can identify examples that possess these qualities that also lead to higher performance gains.

6 Conclusion

In this work, we proposed a Monte Carlo-based sampling method to investigate previous guidance regarding number of examples to use and one- vs. zero-shot settings. We further investigated whether this method could be used to select in-context examples, finding a performance vs. robustness tradeoff.

7 Limitations

Our results are reported on models up to 13B parameters due to constraints posed by our available computational resources. We acknowledge this as a limitation, however, as our results are consistent across the model sizes we utilized, we believe our results should generalize to larger models.

Acknowledgments

We thank the reviewers for their helpful feedback. This research was supported in part by NSF III #2007492.

References

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

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. 2020a. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020b. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Javier Castro, Daniel Gómez, and Juan Tejada. 2009. Polynomial calculation of the shapley value based on sampling. *Computers & Operations Research*, 36(5):1726–1730.

Ting-Yun Chang and Robin Jia. 2023. Data curation alone can stabilize in-context learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8123–8144, Toronto, Canada. Association for Computational Linguistics.

Jiuhai Chen, Lichang Chen, Chen Zhu, and Tianyi Zhou. 2023a. How many demonstrations do you need for in-context learning? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11149–11159, Singapore. Association for Computational Linguistics.

Yanda Chen, Chen Zhao, Zhou Yu, Kathleen McKeown, and He He. 2023b. On the relation between sensitivity and accuracy in in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 155–167, Singapore. Association for Computational Linguistics.

William Dolan, Chris Quirk, Chris Brockett, and Bill Dolan. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Amirata Ghorbani and James Zou. 2019. Data shapley: Equitable valuation of data for machine learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2242–2251. PMLR.

Qi Guo, Leiyu Wang, Yidong Wang, Wei Ye, and Shikun Zhang. 2024. What makes a good order of examples in in-context learning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14892–14904, Bangkok, Thailand. Association for Computational Linguistics.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The Winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, volume 46, page 47.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online

and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. 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.

Sasan Maleki, Long Tran-Thanh, Greg Hines, Talal Rahwan, and Alex Rogers. 2013. Bounding the estimation error of sampling-based shapley value approximation with/without stratifying. *CoRR, abs/1306.4265*, 2(1).

Irwin Mann and Lloyd S Shapley. 1962. Values of large games. 6: Evaluating the electoral college exactly. *RAND Corp., Santa Monica, CA, USA, Tech. Rep. RM-3158-PR*.

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.

Tai Nguyen and Eric Wong. 2023. In-context example selection with influences. *arXiv preprint arXiv:2302.11042*.

Quora. 2017. [link].

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671, Seattle, United States. Association for Computational Linguistics.

Stephanie Schoch, Ritwick Mishra, and Yangfeng Ji. 2023. Data selection for fine-tuning large language models using transferred shapley values. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 266–275.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

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.

Liang Wang, Nan Yang, and Furu Wei. 2024. Learning to retrieve in-context examples for large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1752–1767, St. Julian's, Malta. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

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.

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

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*.

Jiacheng Ye, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2023. Compositional exemplars for in-context learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 39818–39833. PMLR.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022a. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022b. Active example selection for in-context learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9134–9148, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Model	**Parameters**
Pythia (Biderman et al., 2023)	160M 1.4B
OPT (Zhang et al., 2022a)	350M 1.3B
GPT-Neo (Black et al., 2021)	1.3B 2.7B
LLaMa2 (Touvron et al., 2023)	7B 13B

Table 1: Models used in our experiments. We select a range of different model families and parameter sizes. Parameter range is upper bounded based on available compute.

A Additional Background

In-Context Learning: In-context learning enables pre-trained models to learn an unseen task using a set of exemplars concatenated in the context window. Formally, given a test example x, in-context learning concatenates K demonstration examples to the task instruction I, where $S = \{x_{\pi(i)}, y_{\pi i}\}_{i=1}^{K}$ denotes the example set given some order π.

Monte Carlo Sampling: Within the context of data valuation, the underlying idea of Monte Carlo sampling is to sample random permutations of the data points and iterate from the first to last element in each permutation. Specifically, for p Monte Carlo iterations, a dataset D is randomly permuted. Following, these methods scan from the first element of the permutation to the last element of the permutation and compute the performance of the model at each timestep. In data valuation, Monte Carlo sampling methods are used to calculate the marginal contribution of each data point averaged over a number of permutations.

B Experiment Setup Details

We adapt the Language Model Evaluation Harness package (Gao et al., 2023) to conduct our experiments. All experiments use the package's default prompts for each dataset.

Details on models and datasets used can be found in Table 1 and Table 2, respectively.

Notably, the upper bound of the parameter range for models is due to our resource constraint, as each experiment is run using a single NVIDIA A100

Dataset	Task	#Train	#Val	#Classes
MNLI (Williams et al., 2018)	Natural Language Inference	393k	9.82k	3
MRPC (Dolan et al., 2004)	Paraphrase Detection	3.67k	408	2
QNLI (Wang et al., 2018)	Natural Language Inference	105k	5.46k	2
QQP (Quora, 2017)	Paraphrase Detection	364k	40.4k	2
RTE [1]	Textual Entailment	2.49k	277	2
SST-2 (Socher et al., 2013)	Sentiment Analysis	67.3k	872	2
WNLI (Levesque et al., 2011)	Natural Language Inference	635	71	2
Hellaswag (Zellers et al., 2019)	Commonsense Reasoning	39.9k	10k	4

Table 2: Datasets used in our experiments. We use the distributions available from Huggingface (Lhoest et al., 2021), and use the respective validation sets to measure performance.

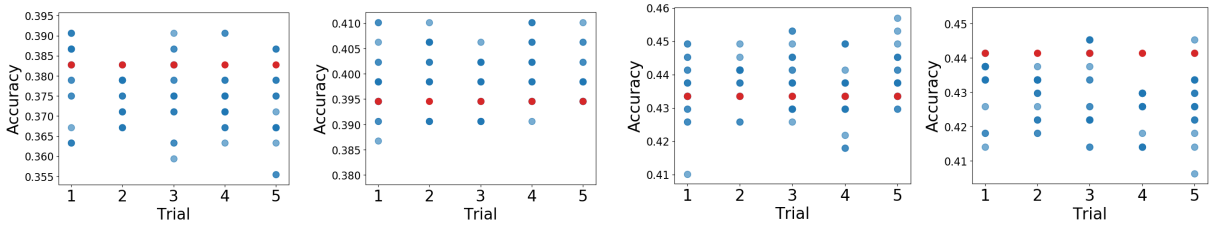
GPU.

For each dataset, we utilize the splits available from Huggingface. As the GLUE benchmark datasets do not have labeled test sets, we use the validation sets for evaluation. Additionally, as we are performing inference after the addition of every example within each permutation, we follow a protocol from prior work and sub-sample 256 instances from the validation set to control inference overhead (Lu et al., 2022).

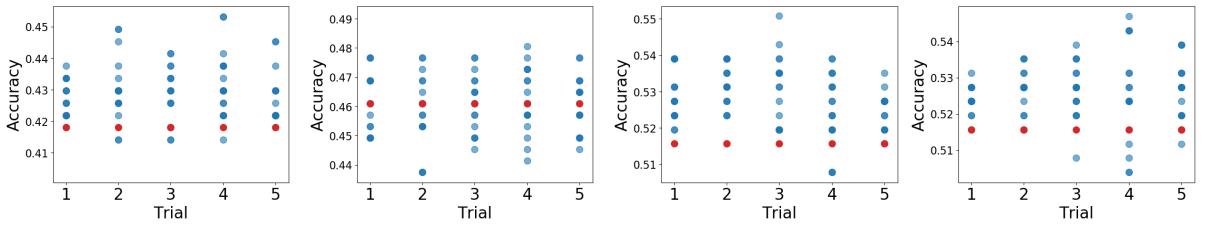
C Additional results

This appendix contains:

- Additional scatter plots (section 5.1)
- Single trial averages overlaying individual permutations (section 5.1)

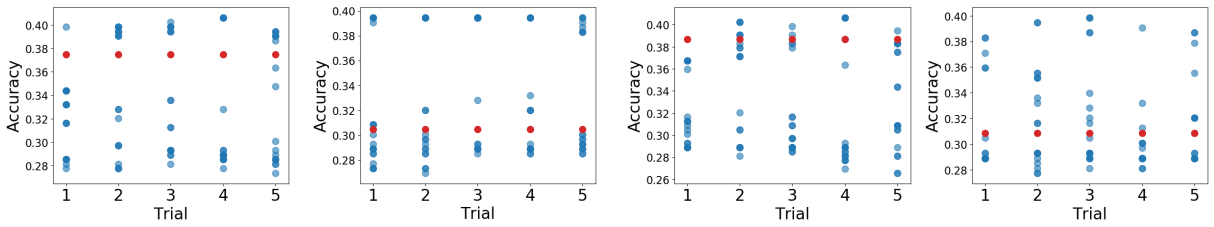


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

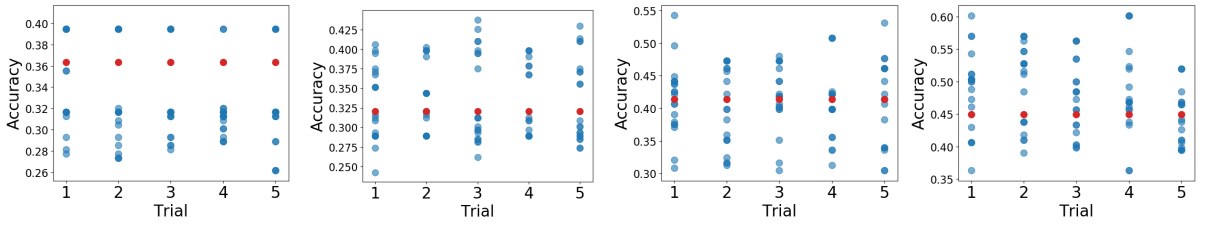


(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 5: One-shot in-context learning performance on the Hellaswag dataset across 5 trials.

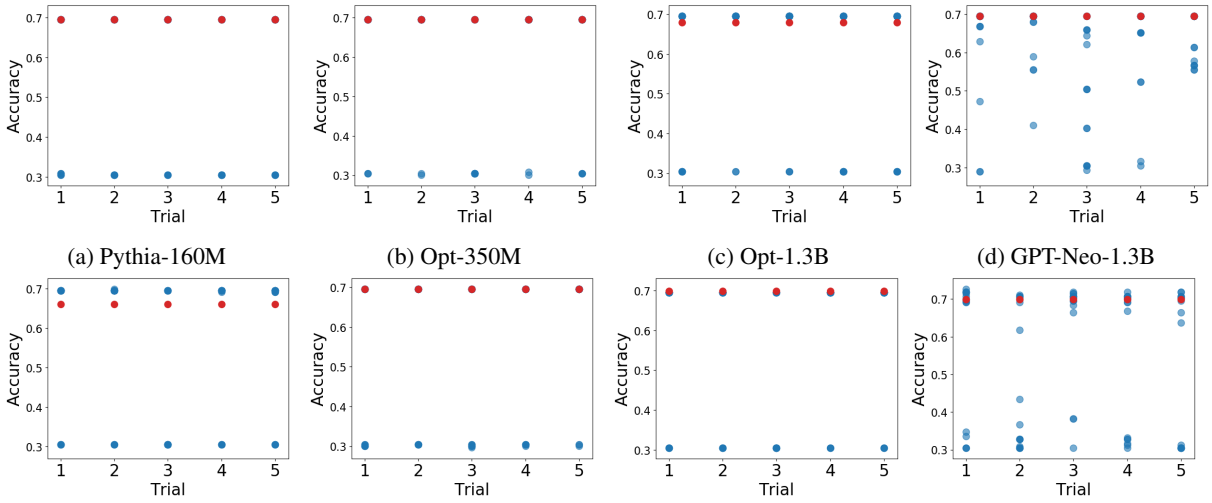


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B



(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

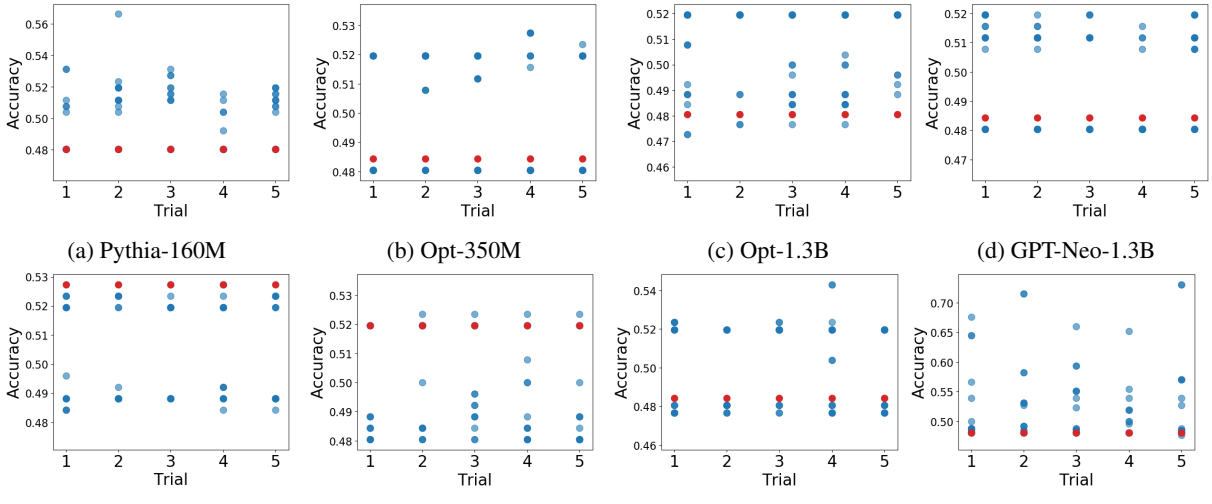
Figure 6: One-shot in-context learning performance on the MNLI dataset across 5 trials. Each blue point represents the accuracy using the first in-context example of a permutation, with 20 permutations per trial. The red points indicate the zero-shot performance of the model. Results indicate that zero-shot settings can outperform one-shot settings, dependent upon the selected in-context example.



(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

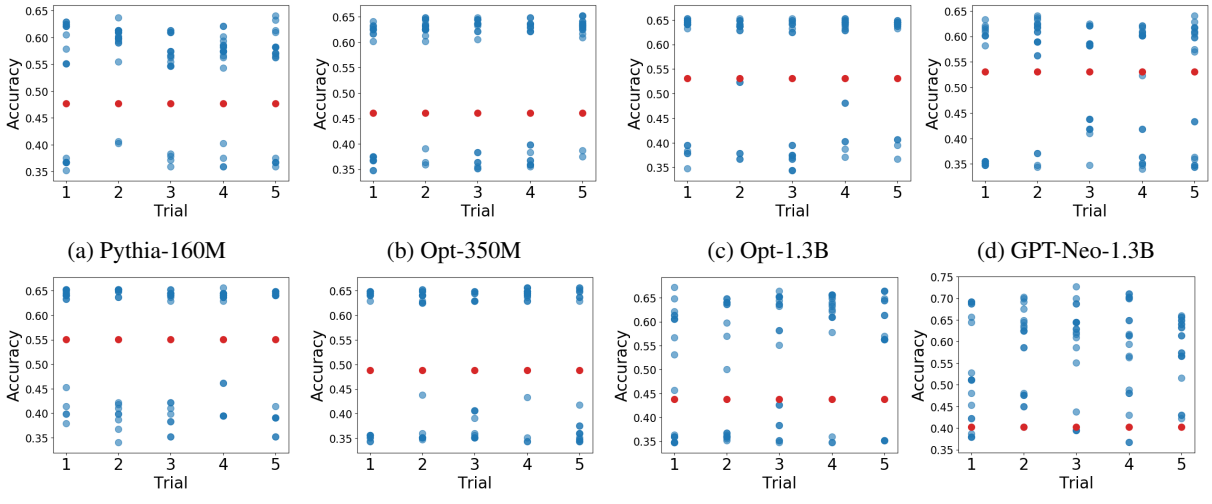
Figure 7: One-shot in-context learning performance on the MRPC dataset across 5 trials.



(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

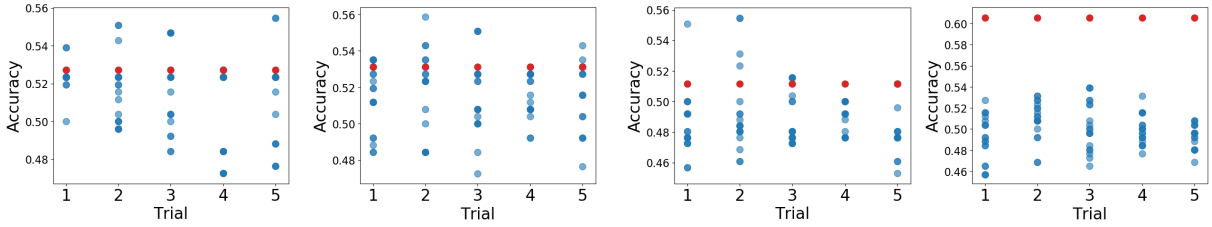
Figure 8: One-shot in-context learning performance on the QNLI dataset across 5 trials.



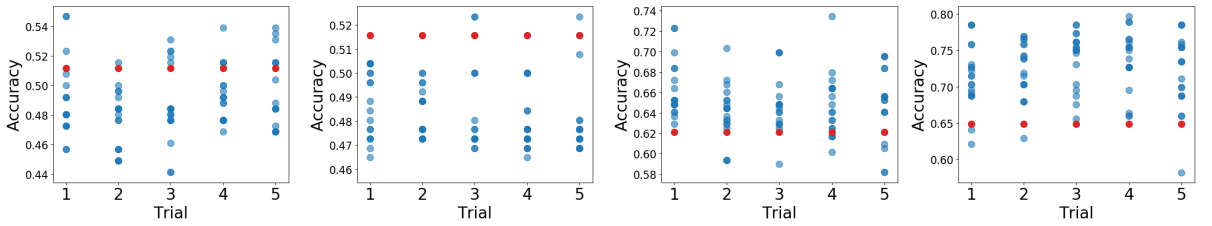
(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 9: One-shot in-context learning performance on the QQP dataset across 5 trials.

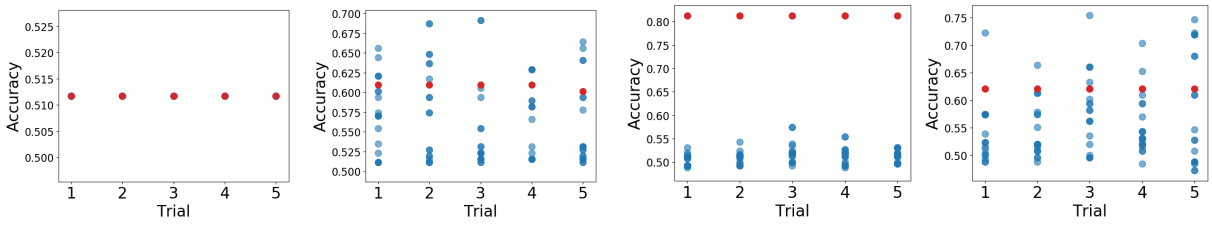


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

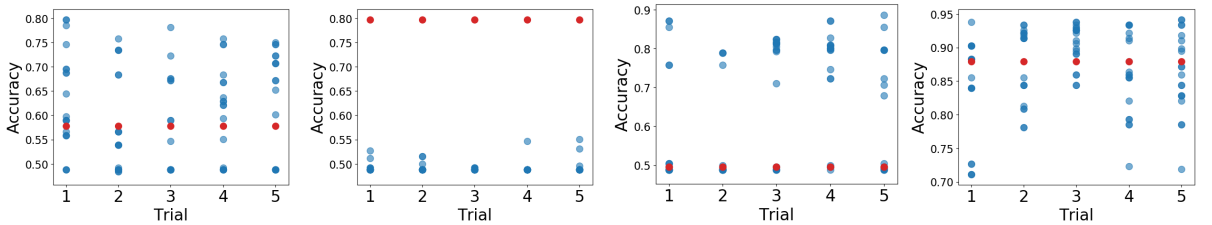


(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 10: One-shot in-context learning performance on the RTE dataset across 5 trials.

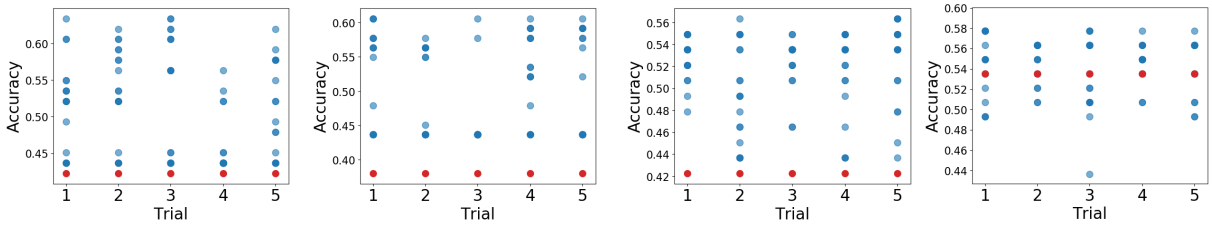


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

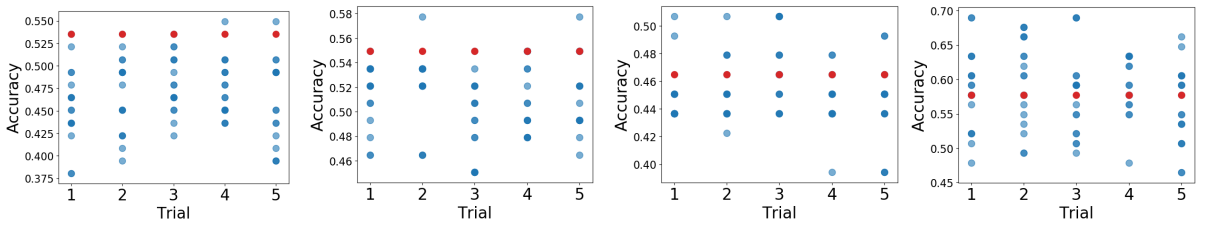


(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 11: One-shot in-context learning performance on the SST-2 dataset across 5 trials.

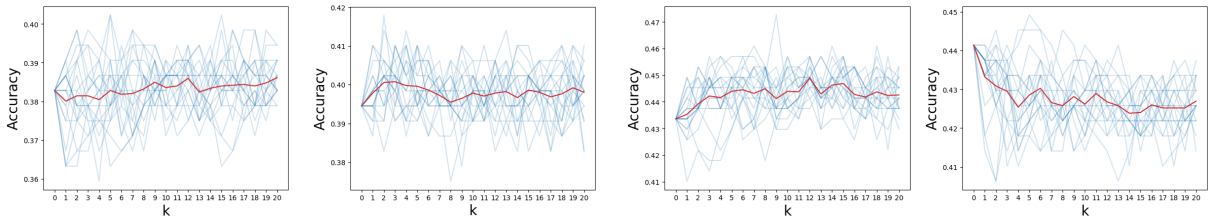


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

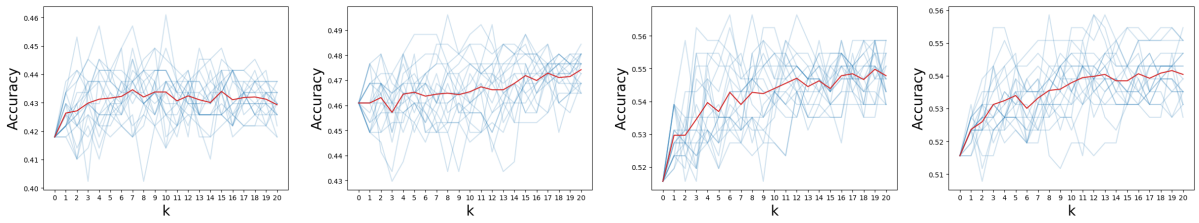


(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 12: One-shot in-context learning performance on the WNLI dataset across 5 trials.

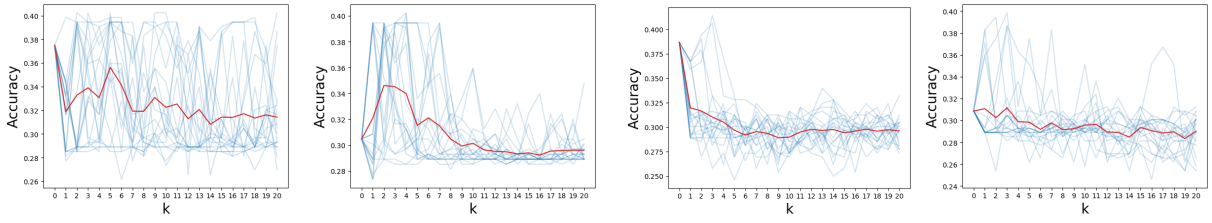


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

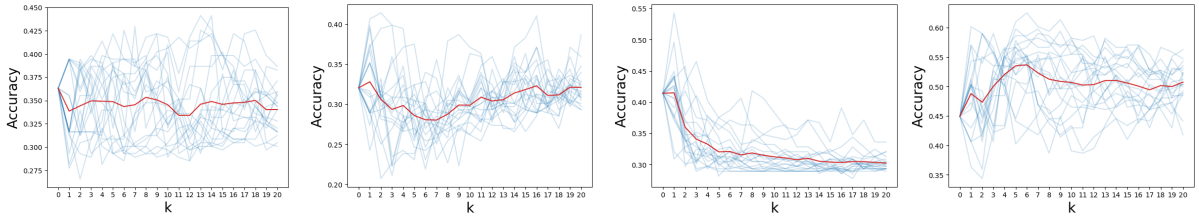


(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 13: Performance of each model on Hellaswag dataset. In each plot, the red line indicates the averages of all permutations for one trial, overlaying blue lines for individual permutations.

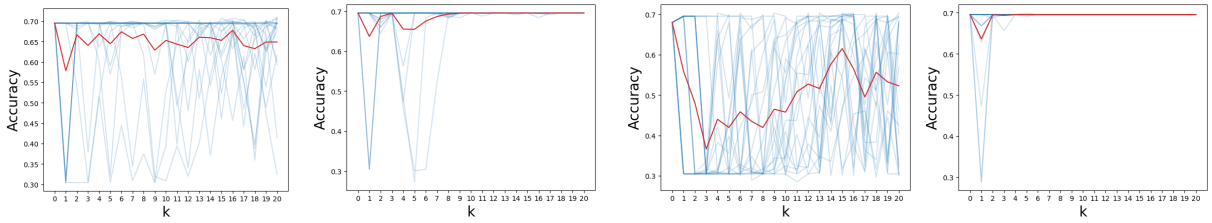


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

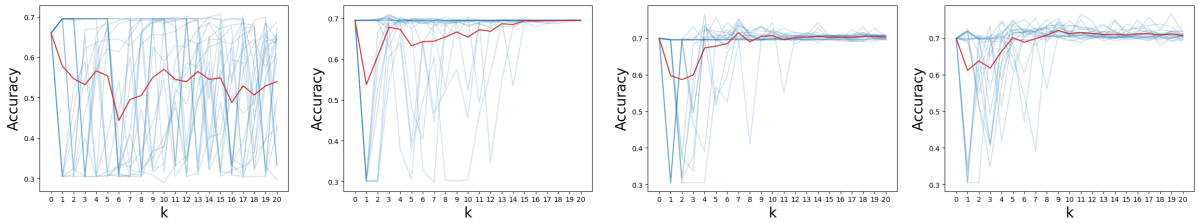


(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 14: Performance of each model on MNLI dataset. In each plot, the red line indicates the averages of all permutations for one trial, overlaying blue lines for individual permutations.

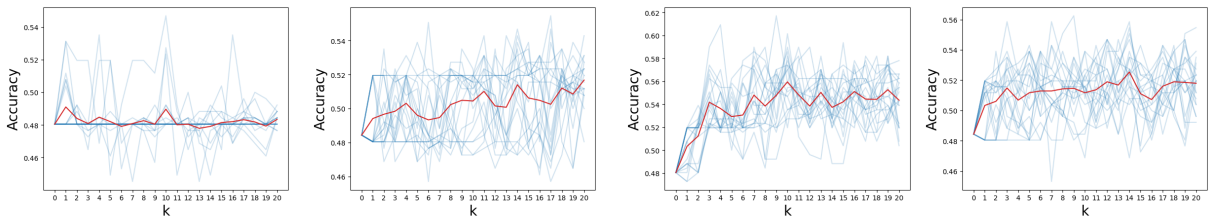


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

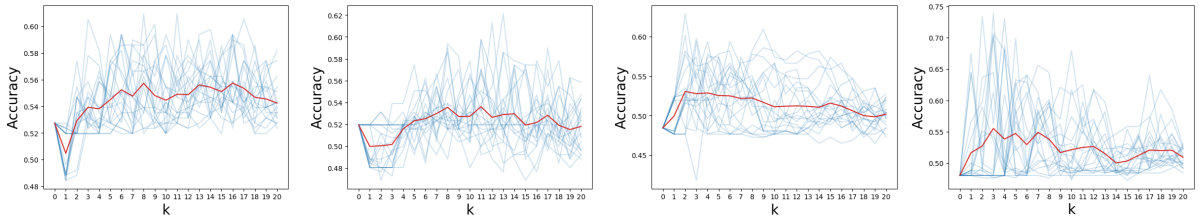


(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 15: Performance of each model on MRPC dataset. In each plot, the red line indicates the averages of all permutations for one trial, overlaying blue lines for individual permutations.

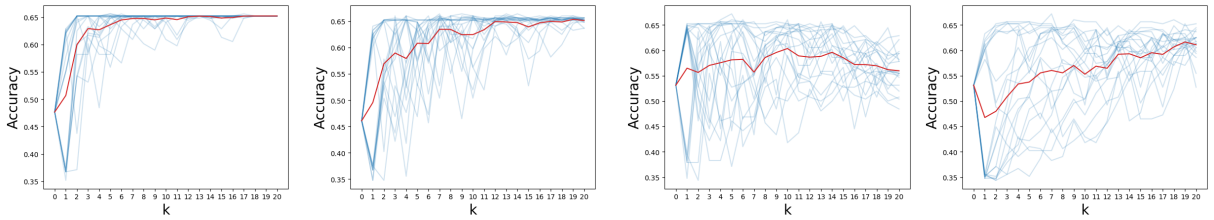


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

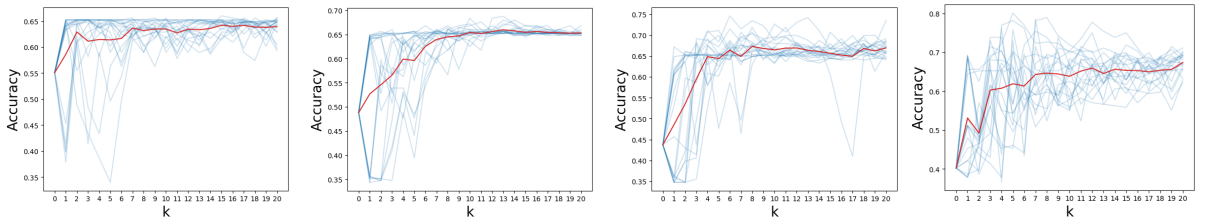


(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 16: Performance of each model on QNLI dataset. In each plot, the red line indicates the averages of all permutations for one trial, overlaying blue lines for individual permutations.

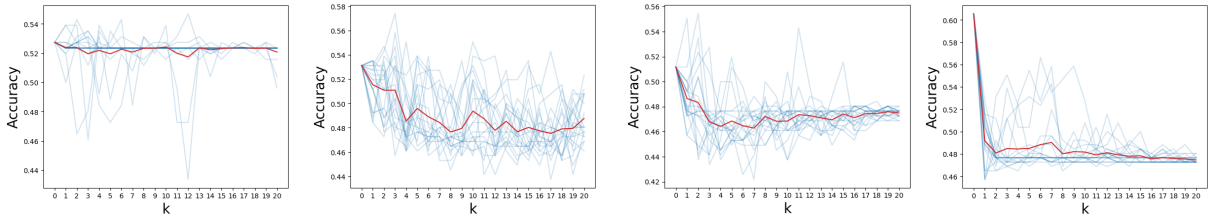


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

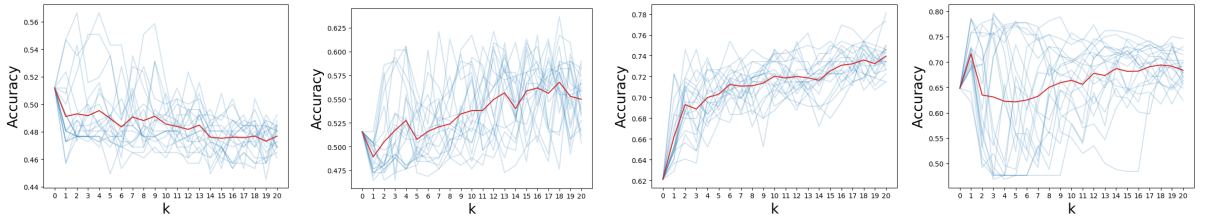


(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 17: Performance of each model on QQP dataset. In each plot, the red line indicates the averages of all permutations for one trial, overlaying blue lines for individual permutations.

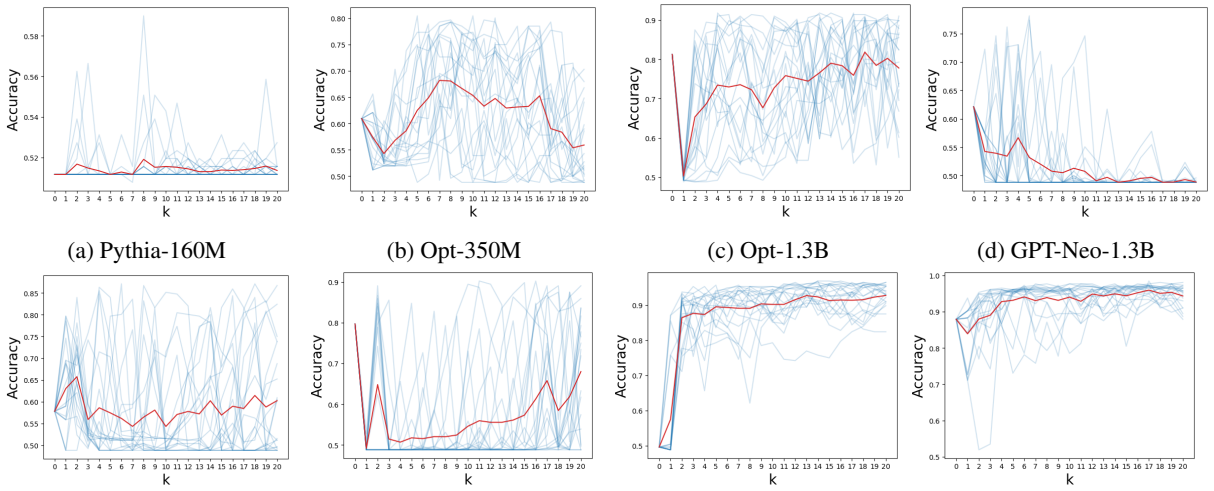


(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B



(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

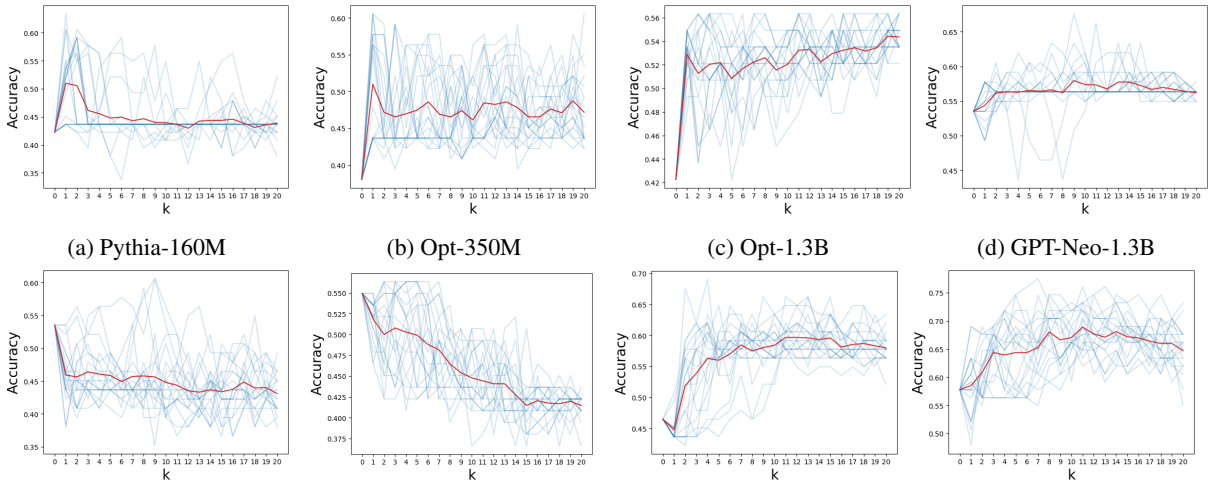
Figure 18: Performance of each model on RTE dataset. In each plot, the red line indicates the averages of all permutations for one trial, overlaying blue lines for individual permutations.



(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 19: Performance of each model on SST-2 dataset. In each plot, the red line indicates the averages of all permutations for one trial, overlaying blue lines for individual permutations.



(a) Pythia-160M (b) Opt-350M (c) Opt-1.3B (d) GPT-Neo-1.3B

(e) Pythia-1.4B (f) GPT-Neo-2.7B (g) Llama2-7B (h) Llama2-13B

Figure 20: Performance of each model on WNLI dataset. In each plot, the red line indicates the averages of all permutations for one trial, overlaying blue lines for individual permutations.