An Exploration of Methods for Zero-shot Transfer in Small Language Models

Alon Albalak[1], Akshat Shrivastava[2], Chinnadhurai Sankar[2], Adithya Sagar[2], Mike Ross[2]
[1]University of California, Santa Barbara [2]Meta AI
`alon_albalak@ucsb.edu`

Abstract

Multi-task learning (MTL), instruction tuning, and prompting have recently been shown to improve the generalizability of large language models to new tasks. However, the benefits of such methods are less well-documented in smaller language models, with some studies finding contradictory results. In this work, we explore and isolate the effects of (i) model size, (ii) general purpose MTL, (iii) in-domain MTL, and (iv) instruction tuning for models with fewer than 500 million parameters. Our experiments demonstrate that general purpose MTL improves performance by 31% on average, with further in-domain MTL improving performance by an additional 37.6%. Contradictory to prior works on large models, we find that instruction tuning provides a modest 2% performance improvement for small models.

1 Introduction

Many recent works have demonstrated the benefits of prompting for large language models (see Liu et al. [2022] for an extensive survey). The utilization of prompts has further expanded into the use of demonstrations, examples, and task instructions, all of which have been shown to improve the generalization of language models to unseen tasks [Wei et al., 2022a, Mishra et al., 2021, Ouyang et al., 2022]. Studies on utilizing prompts have also shown that as model sizes scale up, the generalization abilities of a model also increase [Brown et al., 2020, Lester et al., 2021, Min et al., 2022]. However, utilizing models on the hundred-billion parameter scale is not accessible for most researchers and practitioners. Additionally, some use cases for language models, such as conversational agents, may have strict requirements for memory and latency, reducing the possible use cases for advances in prompting methods.

Similar efforts have demonstrated the benefits of task instructions in the dialogue domain [Gupta et al., 2022]. However, some findings have been contradictory across studies. For example, Wei et al. [2022a] found that models with fewer than 8 billion parameters see decreases in generalization capabilities when training with instructions, whereas Gupta et al. [2022] finds consistent gains in models with 3 billion and fewer parameters. To conflate these results further though, Gupta et al. [2022] only consider 2 situations: when inputs include prompts and instructions, or if inputs include no prompt and no instruction at all.

Simultaneously with the emergence of prompting, the explicit multi-task learning (MTL) paradigm emerged, with works such as Muppet [Aghajanyan et al., 2021] or T0 [Sanh et al., 2022] and their variants. Explicit MTL has been demonstrated as a means of improving the downstream performance of pre-trained language models in data-constrained settings. However, many prior studies of explicit MTL also do not consider models smaller than the billion parameter scale.

In this work we bridge the gap between previous studies by exploring the effects of a variety of factors on the zero-shot generalizability of modestly sized language models (<500 million parameters). Specifically, we run experiments to find the effects of: (i) model size, (ii) general purpose MTL, (iii)

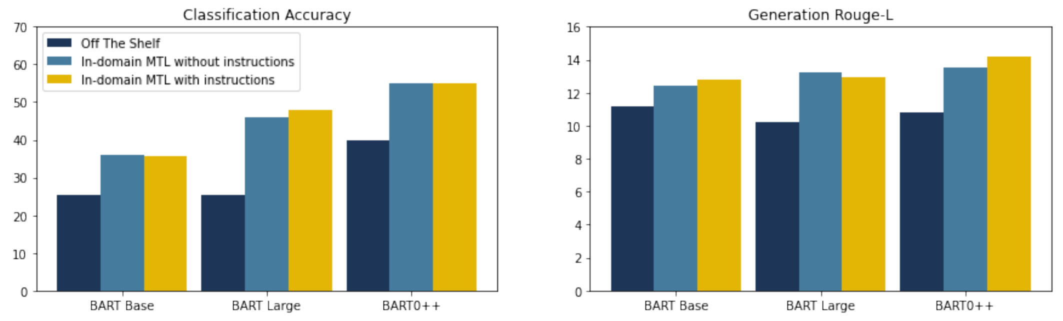

Figure 1: Average model performance on 10 zero-shot classification tasks (left) and 8 zero-shot generation tasks (right) comparing pre-trained models (Off the shelf) with models explicitly multi-task trained on in-domain data with and without instructions. BART0++ is a BART-Large model which has been explicitly multitask trained on PromptSource [Bach et al., 2022] in the same fashion as T0++ Sanh et al. [2022] and demonstrates the effect of explicit multi-task training prior to in-domain training.

in-domain MTL, and (iv) instruction tuning. Additionally, to better understand the sensitivity of models to instruction phrasing, we analyze variations in performance across task instructions.

In this study, we find that **(1)** In-domain multi-task learning (MTL) gives the largest improvements to generalizability, up to 80% increased performance, and 37.6% on average across all models **(2)** Increasing model size alone has little effect on generalization, but when combined with in-domain MTL leads to double the (already strong) performance improvement of in-domain MTL **(3)** General purpose MTL can provide large gains (57% improvement) for downstream tasks which closely resemble the MTL tasks, but still provides modest gains (5%) even for tasks which are more dissimilar **(4)** Instruction tuning during in-domain MTL provides modest gains of just over 2% performance, regardless of model size.

2 Preliminaries

Why should we study small models? Previous studies have shown that trends in large language models (>1 billion parameters) do not hold for smaller language models [Wei et al., 2022b]. For this reason, it is crucial that we must empirically find the trends that occur in smaller models and cannot rely on studies of larger models. Additionally, for situations with latency and memory limitations, small models may be the only option. In particular, we study zero- and few-shot performance on dialogue tasks, a sample domain in which reducing latency and memory usage are of high importance.

What are prompting methods? In this study, we convert all tasks to a sequence-to-sequence format, allowing for a single generative model to perform all tasks [Raffel et al., 2020]. By treating all tasks as sequence-to-sequence, we can also include textual prompts as part of the text input. In this work, we focus on two types of prompts: answer templates and instructions. First, **answer templates** are a string of text added to the end of the input sequence that specifies the task and which allows the model to solve the task by filling in the template in natural language [Liu et al., 2022]. This is in contrast to more simple prompts which only specify the task by including an identifier (eg. "cola sentence" for linguistic acceptability, or "topic" for topic classification) [Raffel et al., 2020, Albalak et al., 2022]. Second, we also consider **instructions**, which are generally added at the beginning of the input sequence and describe the task in natural language. For example, an instruction for document grounded generation is "Read the dialogue and the document text to generate a response."

How is explicit multi-task learning (MTL) used? Explicit MTL has emerged as a strong paradigm for eliciting zero-shot generalization in large language models [Sanh et al., 2022]. In this work we consider 2 types of MTL: general purpose and in-domain. Specifically, general purpose MTL consists of training across a wide variety of tasks and domains, whereas in-domain MTL consists of training across a variety of tasks that all occur within a domain. In this work, we focus on the dialogue domain.

3 Experiments

Data For this study, we utilize 46 annotated tasks from the Instructdial dataset [Gupta et al., 2022]. Each task contains between 3 and 10 instructions, with 4.4 instructions on average across all tasks. For our zero-shot experiments, we use 3 splits of train/test tasks, where each split contains 40 training tasks and 6 test tasks. Tasks are divided into classification and generation tasks, where classification tasks are evaluated on accuracy and generation tasks are evaluated by Rouge-L scores. The full list of tasks and full information on train/test splits can be found in the section A of the Appendix.

Models In our experiments, we utilize 3 variants of the BART encoder-decoder model [Lewis et al., 2020]: BART-Base, BART-Large and BART0++ [Lin et al., 2022]. BART0++ is a BART-Large that has been explicitly multi-task trained on PromptSource [Bach et al., 2022] in the same fashion as T0++ [Sanh et al., 2022].[1]

Experimental Setup To study the effects of (i) model size, (ii) general purpose MTL, (iii) in-domain MTL, and (iv) instruction tuning, we run a series of experiments. In order to measure the effect of (i) model size, we compare performance between BART-Base (139 million parameters) and BART-Large (406 million parameters). To measure the effect of (ii) general purpose MTL, we compare performance between BART-Large and BART0++. To study the effect of (iii) in-domain MTL, we train and test each model on all 3 of the data splits and compare against an off-the-shelf version of each model that is directly tested on each split without any in-domain MTL. These models utilize only an answer template without access to any instructions. To measure the effect of (iv) in-domain MTL with instructions, we train and test each model on all 3 data splits and include instructions in addition to the answer template in the prompt. All experiments were repeated with 3 random seeds, reported scores are means, and standard deviation is reported where appropriate.[2]

4 Findings

Figure 1 shows the average model performance divided into classification and generation tasks. Figure 2 shows the absolute scores for all models and methods on all 18 zero-shot tasks.

Effects of Model Size When comparing the average performance of off-the-shelf versions of BART-Base vs. BART-Large, we find nearly identical performance across classification tasks, and slightly better performance for BART-Base (11.2 vs. 10.2 Rouge-L) on generation tasks. However, the benefits of model size are demonstrated once the models have been further trained using in-domain MTL (Figure 1). We find that with in-domain MTL the base model improves its average score by 6.5. but the large model doubles that improvement, increasing it's score by 13.3 averaged across all tasks.

Effects of General Purpose Multi-task Learning When comparing the performance of BART-Large vs. BART0++ we see improvements on 14/18 tasks, and an average absolute improvement of 14.5 accuracy (57.1% improvement) on classification tasks (Figure 1 left) and more modest improvement of 0.6 Rouge-L (5%) on generation tasks (Figure 1 right). This large discrepancy is likely due to the distribution of tasks in the P3 dataset Bach et al. [2022] used to train BART0++, which consists almost entirely of classification tasks with only summarization as a generation task. Figure 2 shows that, indeed, an off-the-shelf BART0++ outperforms all other methods on summarization, including in-domain MTL.

Effects of In-Domain Multi-Task Learning We find that in-domain MTL (without instructions) contributes the largest portion to the final generalization ability of each model. As shown in Figure 1, BART-Large gets the most benefit with gains of 20.4 points in accuracy (80% improvement) on classification tasks and 3 Rouge-L (29.3% improvement) for generation tasks. Bart-Base gets 41.8% and 11.5% relative improvements on classification and generation tasks, respectively, and BART0++ gets 37.7% and 25.3% relative improvements on classification and generation, respectively. Collectively, this experiment and the previous experiments on general purpose MTL demonstrate the importance of matching both the domain and the task distribution during MTL to the downstream

[1]All pre-trained models were downloaded from the HuggingFace Transformers library.

[2]Further training details on hyperparameters are in section C of the Appendix

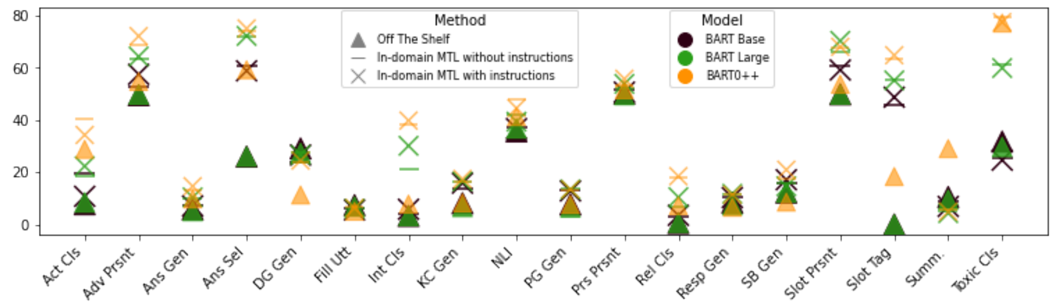

Figure 2: Absolute scores on 18 zero-shot tasks. Full task names are found in section B of the Appendix.

tasks and domain of interest. Additionally, as previously mentioned, in-domain MTL combined with the increased capacity of a larger model shows even greater improvements.

Effects of Instruction Tuning on In-Domain Multi-Task Learning Finally, we compare the performance of in-domain MTL with and without instructions. The benefits of instruction tuning on small models is less prominent than the three previous variables, but is overall still beneficial. Figure 1 shows that BART-Base improves by 3% on generation tasks, but loses 1% accuracy on classification. To the contrary, BART-Large improves by 4% on classification tasks, and loses 2% Rouge-L on generation tasks. Interestingly, BART0++ sees no difference in performance on classification tasks and improves by 5% on generation tasks. These results run counter to those of Wei et al. [2022a], which found that instruction tuning can degrade performance of models with fewer than 8 billion parameters by about 10%. This is likely partly due to the in-domain nature of the instructions utilized in our experiments (all instructions are related to dialogue), suggesting that future works on instruction tuning for small models should focus on (1) domain-specific wording used in instructions, and (2) expanding the number of domains included in instruction sets to see more general benefits.

Findings on Sensitivity to Instructions To better understand the importance of wording and draw insights, we take a closer look at the tasks which had highest variance across instructions. First, we find that Answer selection is the task with highest variance (BART-Base has lowest score of 27.3 and highest score of 63) and find that the three worst performing instructions include variations of "select an option that can substitute <MASK>". The three instructions including this phrase average an accuracy of 39, while the remaining 7 instructions lead to an average accuracy of 60.1. This large discrepancy is likely connected to the unnaturalness of the <mask> token being used in the instruction, and that it is unlikely to have appeared in the BART pre-training corpus, and only appears in 2/46 tasks in our in-domain dataset. The other task which utilizes the <mask> token is the "Fill-in the Missing Utterance" task, which also achieves very poor performance across all models and methods (with and without instructions). This is a strong reminder that to create generalizability in language models, it is crucial to match the downstream task to the pre-training data.

Next, we analyze individual instruction words which most frequently give better than mean performance (see Figure 3 for full results). Interestingly, we find that "return" (as used in "return a response to the conversation") almost always leads to better than average performance (7/8 occurances for BART-Base and Large, and 8/8 for BART0++), although it only occurs in 3 tasks, and 8 instructions.

Finally, we look at the standard deviation between instructions, averaged across all tasks and find very little difference between models, with slightly increasing variation as models get larger, and are pretrained (BART-Base: 0.848, BART-Large: 0.867, BART0++: 0.882). At first glance, this seems to suggest that BART-Base is most robust to wording in instructions, but this is actually due to the smaller number of tasks which BART-Base can meaningfully perform, as seen in Figure 2.

References

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. Muppet: Massive multi-task representations with pre-finetuning. In *Proceedings of the 2021*

Conference on Empirical Methods in Natural Language Processing, pages 5799–5811, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.468. URL `https://aclanthology.org/2021.emnlp-main.468`.

Alon Albalak, Yi-Lin Tuan, Pegah Jandaghi, Connor Pryor, Luke Yoffe, Deepak Ramachandran, Lise Getoor, Jay Pujara, and William Yang Wang. Feta: A benchmark for few-sample task transfer in open-domain dialogue, 2022. URL `https://arxiv.org/abs/2205.06262`.

Stephen H. Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-David, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Alan Fries, Maged S. Al-shaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Xiangru Tang, Mike Tian-Jian Jiang, and Alexander M. Rush. Promptsource: An integrated development environment and repository for natural language prompts, 2022.

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. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

Prakhar Gupta, Cathy Jiao, Yi-Ting Yeh, Shikib Mehri, Maxine Eskenazi, and Jeffrey P. Bigham. Improving zero and few-shot generalization in dialogue through instruction tuning, 2022. URL `https://arxiv.org/abs/2205.12673`.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.243. URL `https://aclanthology.org/2021.emnlp-main.243`.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, 2020.

Bill Yuchen Lin, Kangmin Tan, Chris Miller, Beiwen Tian, and Xiang Ren. Unsupervised cross-task generalization via retrieval augmentation. *ArXiv*, abs/2204.07937, 2022.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, aug 2022. ISSN 0360-0300. doi: 10.1145/3560815. URL `https://doi.org/10.1145/3560815`. Just Accepted.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work?, 2022. URL `https://arxiv.org/abs/2202.12837`.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. Natural instructions: Benchmarking generalization to new tasks from natural language instructions. *ArXiv*, abs/2104.08773, 2021.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155, 2022.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

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. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=9Vrb9D0WI4.

Jason Wei, Maarten Paul Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew Mingbo Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. 2022a. URL https://openreview.net/forum?id=gEZrGCozdqR.

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. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022b. URL https://openreview.net/forum?id=yzkSU5zdwD. Survey Certification.

A Training/Testing Data

The full list of tasks is:

Act Classification, Act Generation, Advice Generation, Advice Present, Answer Generation, Answer Selection, Begins-with Controlled Generation, Belief State Generation, Count Response Words, Database-based Generation, Deal Present, Dialfact Classification, Document Grounded Generation, Edit Generation, Emotion Generation, Emotion Tagging, Ends-with Controlled Generation, Evaluation-Binary, Evaluation-Ranking, Fill-in the Missing Utterance, Find the Incoherent Utterance, Graph-based Generation, Intent Classification, Intent Present, Keyword Controlled Generation, Knowledge Grounded Generation, Natural Language Inference, Non-Toxic Feedback Generation, Persona Grounded Generation, Persuasion Generation, Persuasion Present, Persuasion Strategy, Question Generation, Recovery Generation, Relation Classification, Relation Present, Response Generation with n Words, Response Generation, Schema-based Generation, Slot Present, Slot Tagging, Slot-Value Generation, Summarization, Target Controlled Generation, Toxic Response Classification.

The first data split uses the following 6 tasks for testing, and the remainder for training:

- Act Classification
- Document Grounded Generation
- Intent Classification
- Keyword Controlled Generation
- Relation Classification
- Slot Present

The second data split uses the following 6 tasks for testing, and the remainder for training:

- Answer Selection
- Natural Language Inference
- Persuasion Present
- Schema-based Generation
- Slot Tagging

- Summarization

The third data split uses the following 6 tasks for testing, and the remainder for training:

- Advice Present
- Answer Generation
- Fill-in the Missing Utterance
- Persona Grounded Generation
- Response Generation
- Toxic Response Classification

B Task Abbreviations

- Act Cls - Act Classification
- Adv Prsnt - Advice Present
- Ans Gen - Answer Generation
- Ans Sel - Answer Selection
- DG Gen - Document Grounded Generation
- Fill Utt - Fill-in the Missing Utterance
- Int Cls - Intent Classification
- KC Gen - Keyword Controlled Generation
- NLI - Natural Language Inference
- PG Gen - Persona Grounded Generation
- Prs Prsnt - Persuasion Present
- Rel Cls - Relation Classification
- Resp Gen - Response Generation
- SB Gen - Schema-based Generation
- Slot Prsnt - Slot Present
- Slot Tag - Slot Tagging
- Summ. - Summarization
- Toxic Cls - Toxic Response Classification

C Training Details

We train all models for a maximum of 3 epochs, and utilize validation based early stopping. To determine the learning rate, we trained each model on a single seed and validate the best learning rate in {1e-5, 5e-5, 1e-4}, then train for 2 additional seeds using the best learning rate. We found for all models that 5e-5 was the best learning rate. For all experiments we use the AdamW optimizer.

D Additional Figures

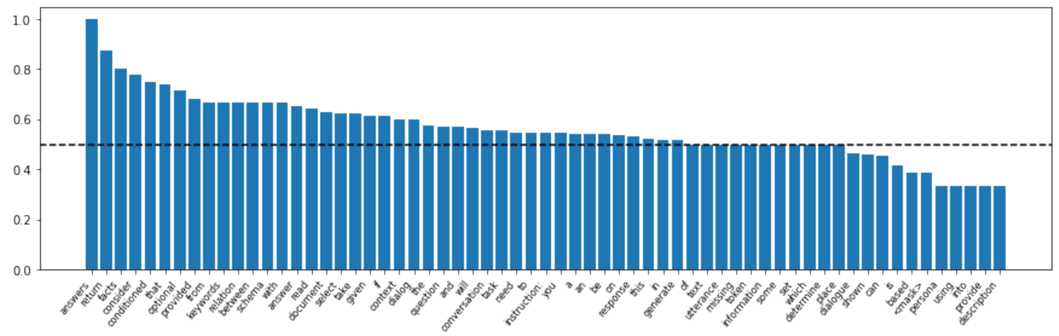

Figure 3: Percentage of occurrences of a word that lead to better than average performance for an instruction. Results calculated from BART-Base model and only includes words that occur is more than 5 instructions.

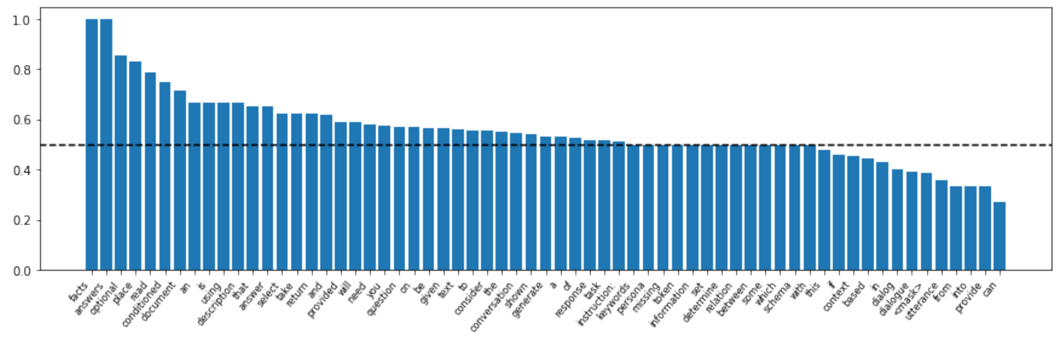

Figure 4: Percentage of occurrences of a word that lead to better than average performance for an instruction. Results calculated from BART-Large model and only includes words that occur is more than 5 instructions.

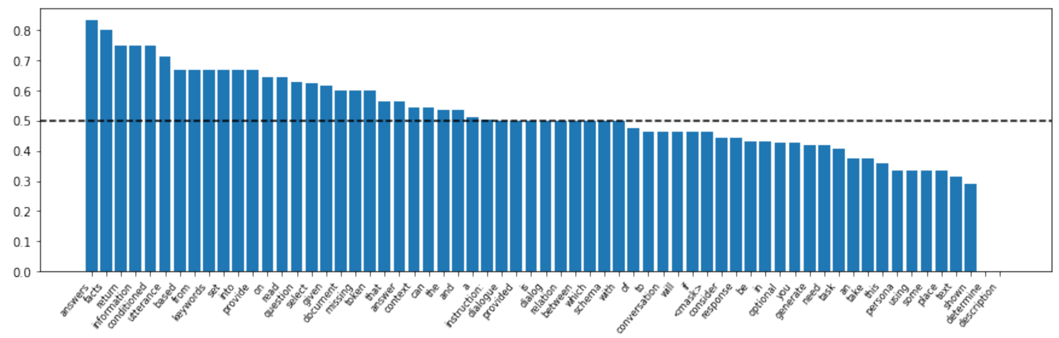

Figure 5: Percentage of occurrences of a word that lead to better than average performance for an instruction. Results calculated from BART0++ model and only includes words that occur is more than 5 instructions.

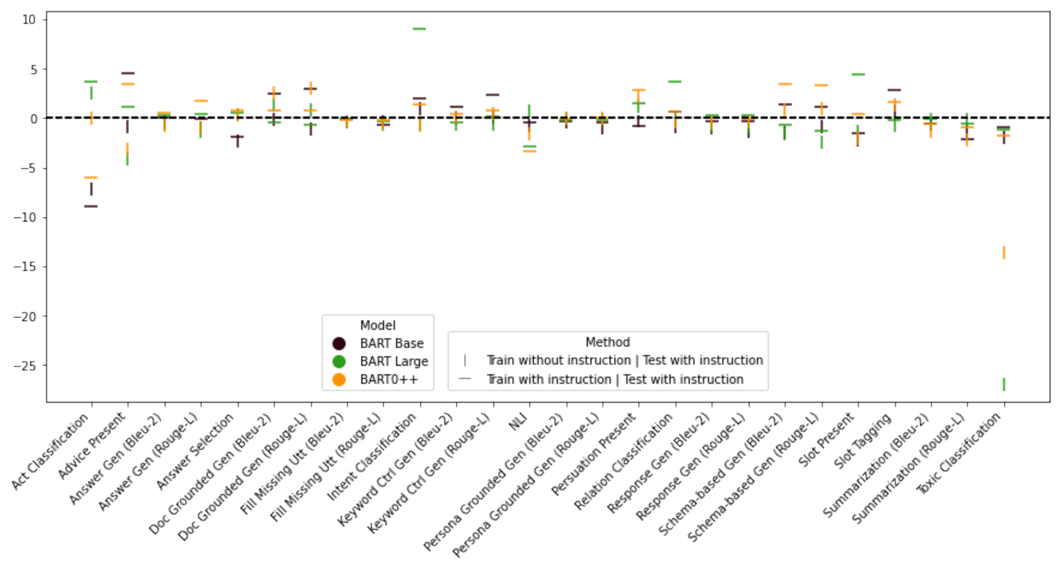

Figure 6: Performance comparison against a baseline model that is trained without instructions. Values shown are the absolute score difference between the designated method and the baseline model.

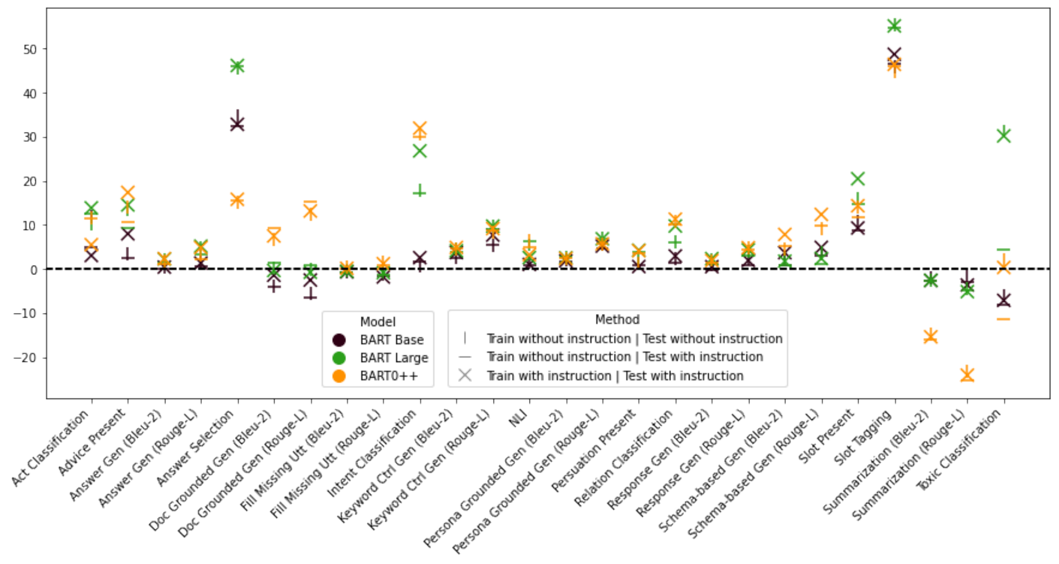

Figure 7: Performance comparison against an off-the-shelf baseline model. Values shown are the absolute score difference between the designated method and the baseline model.