Evaluating Parameter Efficient Learning for Generation

Peng Xu[§], **Mostofa Patwary**[§], **Shrimai Prabhumoye**[§], **Virginia Adams**[§], **Ryan J. Prenger**[§],
Wei Ping[§], **Nayeon Lee**[‡], **Mohammad Shoeybi**[§], **Bryan Catanzaro**[§]
[‡]The Hong Kong University of Science and Technology, [§]NVIDIA
pengx@nvidia.com

Abstract

Parameter efficient learning methods (PERMs) have recently gained significant attention as they provide an efficient way for pre-trained language models (PLMs) to adapt to a downstream task. However, these conclusions are mostly drawn from in-domain evaluations over the full training set. In this paper, we present comparisons between PERMs and finetuning from three new perspectives: (1) the effect of sample and model size to in-domain evaluations, (2) generalization to unseen domains and new datasets, and (3) the faithfulness of generations. Our results show that for in-domain settings (a) there is a cross point of sample size for which PERMs will perform better than finetuning when training with fewer samples, and (b) larger PLMs have larger cross points. For cross-domain and cross-dataset cases, we show that (a) Adapter (Houlsby et al., 2019) performs the best amongst all the PERMs studied here, and (b) it outperforms finetuning if the task dataset is below a certain size. We also compare the faithfulness of generations and show that PERMs can achieve better faithfulness score than finetuning, especially for small training set, by as much as 6%. Finally, we apply Adapter to MT-NLG 530b (Smith et al., 2022) and achieve new state-of-the-art results on Xsum (Narayan et al., 2018) for all ROUGE scores (ROUGE-1 49.17, ROUGE-2 27.20, ROUGE-L 40.98).

1 Introduction

Parameter efficient learning methods (PERMs) serve as potential alternatives to finetuning for adapting and deploying language models in real world scenarios (Ding et al., 2022). They allow users to finetune only a small number of parameters while freezing the rest of the shared parameters of pre-trained language models (PLMs). This is especially important for large language models (e.g. GPT-3 (Brown et al., 2020) and MT-NLG (Smith et al., 2022)) as finetuning the entire model will be very expensive or infeasible due to their model size.

Prefix tuning (Li and Liang, 2021), which is one of the PERMs, draws inspiration from prompting and introduces a small set of continuous vectors as virtual prompts to allow subsequent tokens to attend to, which obtains comparable performance to finetuning in the full data setting. Prompt tuning (Lester et al., 2021) shows the power of scaling PLMs and that tuning only a few extra embeddings is sufficient to achieve similar performance to finetuning the entire 11b T5-XXL (Raffel et al., 2020) model. P-tuning v2 (Liu et al., 2022a) further demonstrates that small PLMs can also achieve comparable results to finetuning with Prefix tuning. Different from adding new parameters through prompts, Adapter (Houlsby et al., 2019) injects trainable parameters through low-rank structure in a skip-connection way. Other PERMs includes LoRA (Hu et al., 2021), Mix-And-Match adapter (He et al., 2021a), Compactor (Karimi Mahabadi et al., 2021), BitFit (Zaken et al., 2022), diff-pruning (Guo et al., 2021) and etc.

Most conclusions about PERMs so far are drawn from their in-domain evaluations over full training samples. To the best of our knowledge, it is not yet investigated (1) how these conclusions apply to different training sizes and model sizes, and (2) how PERMs generalize to unseen domains and new datasets, which are both important aspects for deploying PERMs in real-world applications.

In addition, faithfulness in natural language generation has become an important topic as it is vital to real-world applications. Various efforts are made to systematically measure and mitigate factual errors in many generation tasks, including summarization (Huang et al., 2021) and dialogue generations (Rashkin et al., 2021; Shuster et al., 2021; Dziri et al., 2021; Wu et al., 2021). However, existing work on faithfulness only focuses on faithfulness of finetuning, and the impact of PERMs on

4824

the faithfulness of generation is not yet explored.

In this paper, we provide an in-depth study of PERMs for generation tasks through three important aspects when deploying PERMs in practical applications: (1) in-domain evaluation by scaling both training dataset size and model size of PLMs, (2) cross-domain and cross-dataset generalization, and (3) faithfulness assessment. Two generation tasks are used for evaluation: summarization and dialogue generation. We study four representative methods: P-tuning, Prompt tuning, Prefix tuning, and Adapter, but mainly focus on Prefix tuning and Adapter as our preliminary results show that they are better than the others. Our contributions are summarized as follows: (1) To the best of our knowledge, we present the first comparisons of faithfulness for PERMs. Our experimental results show that PERMs, especially prefix tuning can achieve better faithfulness than finetuning by up to 6%. (2) For in-domain settings, there is always a cross point of sample size for which PERMs will be better than finetuning when training on fewer samples. Larger PLMs have larger cross points. Users need to choose which method to use based on their own training sample size and model size. (3) Compared to finetuning, not all PERMs can easily achieve better cross-domain and cross-dataset scores than finetuning even with 8.3b PLM. Our results show that Adapter is a better method than Prefix tuning on 13 out of 15 comparison settings. (4) New state-of-the-art results on Xsum (Narayan et al., 2018) are obtained by applying Adapter to MT-NLG 530b model.

2 Methodology

We compare the following four PERMs to **finetuning (FT)** using GPT-style models from Megatron-LM (Shoeybi et al., 2019). **(1) Adapter (AP)** adds an extra layer with a bottleneck structure by first projecting input h to a low dimension using trainable weights W_{down} and then projecting up to the original dimension using trainable weights W_{up}. It is incorporated into backbone model in a skip-connection way.

$$Adapter(h) = h + g(hW_{down})W_{up},$$

where g is the activation function. In our case, we insert *Adapter* layer both after the multi-head attention (MHA) and feedforward layer (FFD) of Transformer (Vaswani et al., 2017). **(2) Prefix Tuning (PF)** adds trainable prefix tokens at the beginning

of each transformer block. We follow the implementation of Li and Liang (2021) to replace the keys K, values V of MHA with the concatenation of the trainable prefix weights W_K, W_V and the K, V.

$$K \leftarrow concat([W_K; K])$$
$$V \leftarrow concat([W_V; V])$$

We also add reparameterization trick suggested by Li and Liang (2021). **(3) Prompt Tuning (PT)** adds extra parameters to the embedding layer and uses these trainable embeddings to prompt the input. **(4) P-tuning (Liu et al., 2021b)** adds a prompt encoder to encode pseudo prompts and the encoded representation is used to prompt the input.

3 Experimental Setup

3.1 Datasets

Summarization We use Xsum (Narayan et al., 2018), a widely used summarization dataset, to train and evaluate different methods. It consists of 204,017/11,327/11,333 pairs for the training/validation/test. As Xsum does not divide the dataset based on topics, we follow Li and Liang (2021) to split the Xsum dataset into news articles for training and sports articles for testing. This cross-domain version has 149,115/8,263/2,823 pairs for training/validation/test. For the cross-dataset evaluation, we choose the test set from CNN/Daily Mail (Nallapati et al., 2016). It contains 11,490 samples.

Dialogue We use Wizard of Wazards (WoW) (Dinan et al., 2018) dataset for our dialogue generation task. The modeling of the wizard response is usually composed of two steps: knowledge retrieval and response generation. To simplify the problem, following Rashkin et al. (2021), we ignore the knowledge retrieval step and take the golden knowledge for the response generation. The response of the wizard is then used to train the model. For the cross-dataset evaluation, we use the CMU_DoG (Zhou et al., 2018) dataset. We test our model over all test set dialogue turns except the starting one.

3.2 Metrics

Quality Metrics We use ROUGE-1 (R-1), ROUGE-2 (R-2), ROUGE-L (R-L) (Lin, 2004) scores to evaluate the generations for summarization task as it is well adopted in all summarization tasks. For the dialogue generation task, Dinan et al.

Method	Parameter	R-1	R-2	R-L
P-tuning	72k	33.3	11.2	26.0
PT	154k	32.7	10.8	25.5
PF	5m	35.3	13.5	27.9
AP	5m	**37.7**	**15.3**	**30.1**
FT	357m	**41.6**	**19.2**	**33.8**

Table 1: Xsum results by comparing different methods over 357m GPT model using full dataset. Parameter here counts extra task parameters needed during inference. FT is much better than PERMs for all ROUGE metrics with p-value lower than 0.001 through a t-test. AP and PF is also better than P-tuning and PT with p-value lower than 0.001 through a t-test.

(2018) reports both PPL and unigram F1 scores and better PPL in general gives better F1 scores. Adiwardana et al. (2020) also shows high correlation between PPL and the quality of dialogue based on human evaluations. We therefore choose to report PPL as an indicator of the quality of generated dialogues. In the Results section, if we say "A is better than B", we mean A has a higher ROUGE score for summarization task or/and a lower PPL score for dialogue tasks.

Faithfulness Metrics Following Rashkin et al. (2021), we use a state-of-the-art natural language interference (NLI) model (Roberta trained on MNLI (Liu et al., 2019)) to predict whether a response can be entailed by the given evidence. We evaluate the faithfulness of generated response against the concatenation of dialogue history and the golden knowledge. Entailment score is reported as the ratio of the samples being predicted as entailment from the NLI model. We use factCC (Kryściński et al., 2020) to evaluate the faithfulness for the Xsum as it has the highest Spearman correlation with human evaluations (Pagnoni et al., 2021).

4 Results

4.1 In-domain Results

In-domain evaluations are presented in Table 1. Although Adapter(AP) and Prefix Tuning (PF) are better than prompt tuning and p-tuning, they are still much worse than FT (3.7 lower for R-L). To better understand when PERMs is better than FT, we scale both the training sample size and model size for summarization and dialogue generation task. As PF and AP are much better than other PERMs, we focus on those two methods.

The results on Xsum and WoW are shown in Figure 1a and Figure 1b. Comparing AP with PF,
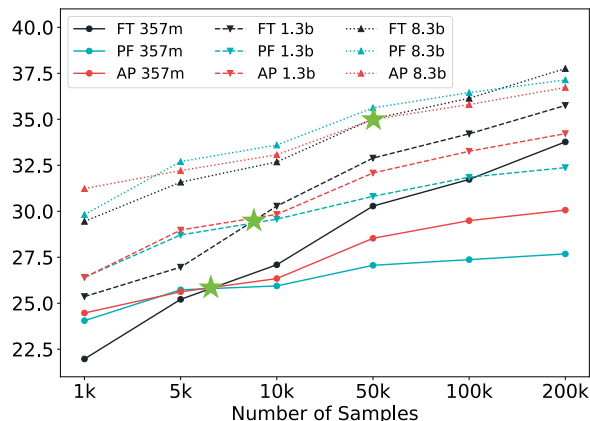
(a) R-L comparison over Xsum test set
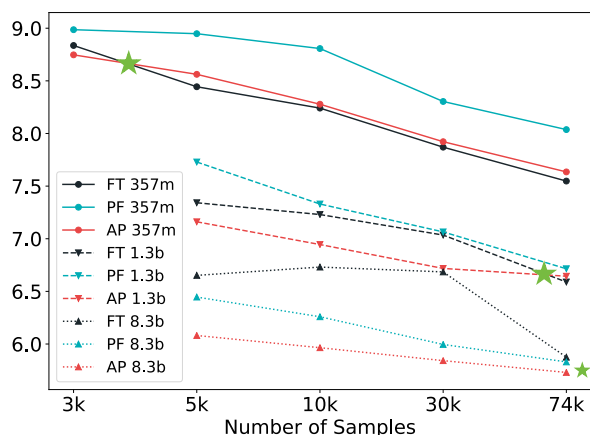
(b) PPL comparison over WoW seen test set

Figure 1: There is a cross point of sample size where PERMs, e.g. AP will be better than FT when training fewer samples. Larger PLMs have larger cross points.

we find that AP is better than PF on 26 out of 31 comparisons. It is also aligned with the conclusion in Ding et al. (2022). This can be attributed to the structural bias of Adapter. The skip-connection structure allows Adapter to add a small deviation to the activation, which makes the optimization of the PLM checkpoint smooth. On the contrary, PF introduces deviations to the keys and values of the self-attention module and therefore greatly varies the activation of each layer. As a result, it takes more efforts for PF to converge. Another phenomenon we observed in Figure 1 is that if we train a 8.3b model with enough training samples (74k for WoW or 200k for Xsum), the performance gap between PF, AP and FT is quite marginal. This suggests us to use PERMs instead of FT to save the cost of deploying 8.3b PLMs.

Comparing FT with AP, we find there is always a cross point of sample size where FT is better than

Method	Parameter	R-1	R-2	R-L
BRIO	568m	49.07	25.59	40.40
T5	11b	48.83	25.96	40.70
MT-NLG	103m	49.17	27.20	40.98

Table 2: Xsum results by comparing MT-NLG AP to other state-of-the-art models: (1) BRIO (Liu et al., 2022b) (2) T5 (Rothe et al., 2021). MT-NLG achieves new state-of-the-art results.

model size	parameters	PF	AP
1.3b	5m	32.76	33.73
1.3b	10m	32.37	34.22
8.3b	5m	37.21	37.71
8.3b	33m	37.14	36.73

Table 3: R-L score for PF and AP. More parameters do not always give better results.

model size	dataset	PF	AP	FT
357m	WoW	8.35	8.01	**7.94**
1.3b	WoW	6.99	6.91	**6.89**
8.3b	WoW	6.06	**5.95**	6.11
357m	Xsum	23.86	**24.55**	23.07
1.3b	Xsum	**28.85**	28.60	27.74

Table 4: Cross-domain evaluation with R-L and PPL over Xsum and WoW trained with full dataset samples.

model size	samples	PF	AP	FT
357m	5k	18.70	**19.91**	17.11
1.3b	5k	19.19	**19.22**	17.97
357m	200k	**17.84**	16.54	14.00
1.3b	200k	15.43	**15.81**	14.20

Table 5: Cross-dataset generalization evaluation over CNN/Daily Mail using R-L. PERMs outperforms FT.

AP. This shows that if we have large number of samples in training set, FT will work better. But if the number of samples for the task are small, AP will be better. Also, this cross point will be larger if we use larger PLMs. For example, the cross point for 1.3b model over Xsum is less than 10k samples, whereas for the 8.3b model, it is 50k samples . This phenomenon can be attributed to that FT can easily overfit when you have large models or few training samples. It motivates us to use AP when you have small dataset or large model to achieve better in-domain performances.

Interestingly, tuning AP with a 8.3b model of only 32m extra parameters over 5k samples achieves much better results than finetuning 357m model over 100k samples. This means more than 90% task specific parameters can be saved for deployment and more than 97% tasks specific samples can be reduced for training by sharing the larger frozen PLMs.

Scaling up to 530b model Since AP gets better performances than other methods, we apply AP to one of the largest GPT model, MT-NLG (Smith et al., 2022). Table 2 shows that by adding 103m parameters to a frozen 530b model, we achieves a new state-of-the-art result on Xsum for all ROUGE scores (R-1 49.17, R-2 27.20, R-L 40.98). It outperforms both strong encoder-decoder models (e.g. T5 (Rothe et al., 2021)), as well as specialized models (e.g. BRIO (Liu et al., 2022b)). This result shows that decoder-only model can still beat encoder-decoder model, but it needs a much larger model size.

We also study the effects of varying parameter sizes for PERMs in Table 3. We found that the score of AP increases for 1.3b model, which suggests the model is under-fitting. He et al. (2021a) observed a similar trend with a similar sized model (700M). On the other hand, the score of PF drops for 1.3b model, which suggests it is overfitting. This difference can be attributed to the way we count the parameters in Table 3. Note that the number of parameters for PF is counted as extra *inference* parameters following Li and Liang (2021), which is different from trainable parameters. For example, PF 1.3b model with 10m extra inference parameters actually contains 80m extra trainable parameters, which is much higher than the 10m shown in the table. Such a large number of trainable parameters will easily make the model overfit for PF and thus leads to the performance drop with more parameters. For the 8.3b model, the scores of both AP and PF drops with more parameters as both of them are overfitting and AP has a more serious overfitting issue there. Table 3 suggests that (1) more parameters do not always help PF or AP, (2) task specific parameters can be further reduced by sacrificing little scores.

4.2 Cross-domain and Cross-dataset Generalization

Table 4 shows cross-domain results over Xsum and WoW and cross-dataset results can be found in Table 5 and Table 6. We find that AP achieves in general better generalization than PF in cross-domain and cross-dataset setting by 13 out of 15

model size	samples	PF	AP	FT
357m	5k	29.7	**26.7**	27.1
1.3b	5k	26.3	**19.7**	21.0
8.3b	5k	16.9	**15.3**	17.5
357m	74k	30.0	27.6	**26.2**
1.3b	74k	24.0	**20.1**	21.0
8.3b	74k	17.8	**16.3**	16.5

Table 6: Cross-dataset generalization evaluation over CMU_DoG using PPL.

model size	samples	PF	AP	FT
357m	5k	**0.815**	0.800	0.751
1.3b	5k	**0.768**	0.749	0.713
8.3b	5k	**0.752**	0.733	0.700
357m	74k	**0.788**	0.767	0.762
1.3b	74k	**0.760**	0.744	0.750
8.3b	74k	**0.721**	0.705	0.720

Table 7: Entailment score for WoW test seen dataset. PF achieves the best score.

model size	samples	PF	AP	FT
357m	200k	**0.252**	0.239	0.232
1.3b	200k	**0.243**	0.241	0.227
8.3b	200k	**0.231**	0.219	0.227

Table 8: FactCC scores for Xsum. PF achieves the best score.

comparisons.

For Xsum, both AP and PF are universally better than FT for cross-domain and cross-dataset setting across different training sample sizes. For WoW, we find the conclusion is a bit different. PF is worse than FT with 7 out of 9 comparison settings in Table 4 and Table 6 while AP has only 3 out of 9. We conjecture this is due to the bias of different tasks as we can see the cross points for Xsum and WoW are also quite different. WoW reflects a more general finding that AP can achieve better cross-domain and cross-dataset scores than FT when PLMs are large enough (e.g. 1.3b) or training samples are small enough (e.g. 5k). In these two cases, AP only tunes a relatively small amount of parameters comparing to FT and therefore is less likely to overfit. It is also aligned with conclusions under in-domain scenario.

4.3 Faithfulness

Table 7 and Table 8 shows the faithfulness evaluation over WoW and Xsum dataset using entailment score and factCC score. The faithfulness score for Xsum is quite low as the dataset contains many unfaithful training samples (Pagnoni et al., 2021). Both of the tables show that PF achieves the best faithfulness score across all model sizes and sample size. However, when increasing the PLM size from 357m to 8.3b, or training samples from 5k to 74k, we see a constant drop of entailment score or factCC score. This can be attributed to (1) both WoW and Xsum have many responses or summaries that contains information external to the evidence (Rashkin et al., 2021; Pagnoni et al., 2021) and (2) larger language models memorize more world knowledge itself (Brown et al., 2020). Therefore, our models will become more unfaithful when they learn from those unfaithful examples or use its embedded knowledge. In such case, PF provides an option for users to sacrifices a little PPL to earn more faithfulness. How to further improve

the faithfulness of PERMs is still an open research problem and we leave it for future work.

5 Conclusion

In this paper, we extensively compare PERMs with finetuning over three main areas: (1) in-domain evaluation by scaling both the sample size and model size (2) cross-domain and cross-dataset generalization (3) faithfulness of generations. For in-domain settings, we find (a) there is a cross point of sample size for which parameter efficient learning will be better than finetuning when training with fewer samples and (b) larger PLMs have larger cross points. This suggests users to choose the method based on their own sample size and model size. Simply apply Adapter to MT-NLG, we achieve new state-of-the-art results on Xsum for all ROUGE scores (ROUGE-1 49.17, ROUGE-2 27.20, ROUGE-L 40.98). Compared to finetuning, not all PERMs can easily achieve better cross-domain and cross-dataset scores than finetuning even with large PLM (e.g. 8.3b). Adapter is a better choice than other PERMs in such cases. Lastly, we provide the first comparison of PERMs over faithfulness of generations and show that Prefix tuning is the best method for faithfulness. We believe our findings will help users better choose and deploy PERMs.

6 Limitations

Our paper have the following limitations. Firstly, we are only able to qualitatively show the cross point when FT is better than AP. We do not derive

a quantitative estimation of the cross point given the model size and the task name. Therefore, the cross point of our paper can be served as a reference only for summarization and dialogue generation when choosing between these methods. (2) Even though we show PF achieves better faithfulness than other methods. We found that when the model is large enough, (e.g. 8.3b) and the dataset is large too (e.g. 74k), PF achieves quite close scores to FT (0.721 vs 0.720). Therefore, it remains a question how to achieve better faithfulness under such setting.

References

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*.

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

Shuyang Cao and Lu Wang. 2021. Cliff: Contrastive learning for improving faithfulness and factuality in abstractive summarization. *EMNLP*.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact aware neural abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904*.

Yue Dong, Shuohang Wang, Zhe Gan, Yu Cheng, Jackie Chi Kit Cheung, and Jingjing Liu. 2020. Multi-fact correction in abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9320–9331.

Nouha Dziri, Andrea Madotto, Osmar Zaiane, and Avishek Joey Bose. 2021. Neural path hunter: Reducing hallucination in dialogue systems via path grounding. *EMNLP*.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. Ppt: Pre-trained prompt tuning for few-shot learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8410–8423.

Demi Guo, Alexander M Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. 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 4884–4896.

Wenjuan Han, Bo Pang, and Ying Nian Wu. 2021. Robust transfer learning with pretrained language models through adapters. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 854–861.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021a. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*.

Ruidan He, Linlin Liu, Hai Ye, Qingyu Tan, Bosheng Ding, Liying Cheng, Jiawei Low, Lidong Bing, and Luo Si. 2021b. On the effectiveness of adapter-based tuning for pretrained language model adaptation. 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 2208–2222.

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 *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Yichong Huang, Xiachong Feng, Xiaocheng Feng, and Bing Qin. 2021. The factual inconsistency problem in abstractive text summarization: A survey. *arXiv preprint arXiv:2104.14839*.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical*

Methods in Natural Language Processing (EMNLP), pages 9332–9346.

Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. 2019. Hallucinations in neural machine translation. *ICLR*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. 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.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Tianyu Liu, Xin Zheng, Baobao Chang, and Zhifang Sui. 2021a. Towards faithfulness in open domain table-to-text generation from an entity-centric view. In *AAAI*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022a. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

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*.

Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. 2022b. Brio: Bringing order to abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2890–2903.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. 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 565–576.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.

Feng Nie, Jin-Ge Yao, Jinpeng Wang, Rong Pan, and Chin-Yew Lin. 2019. A simple recipe towards reducing hallucination in neural surface realisation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2673–2679. ACL.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with frank: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Hannah Rashkin, David Reitter, Gaurav Singh Tomar, and Dipanjan Das. 2021. Increasing faithfulness in knowledge-grounded dialogue with controllable features. 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 704–718.

Clément Rebuffel, Marco Roberti, Laure Soulier, Geoffrey Scoutheeten, Rossella Cancelliere, and Patrick Gallinari. 2022. Controlling hallucinations at word level in data-to-text generation. *Data Mining and Knowledge Discovery*, pages 318–354.

Sascha Rothe, Joshua Maynez, and Shashi Narayan. 2021. A thorough evaluation of task-specific pre-training for summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 140–145, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion

parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. *EMNLP*.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, pages 5986–5995. PMLR.

Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, et al. 2022. On transferability of prompt tuning for natural language processing. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. 2022. Spot: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5039–5059.

Chaojun Wang and Rico Sennrich. 2020. On exposure bias, hallucination and domain shift in neural machine translation. In *2020 Annual Conference of the Association for Computational Linguistics*, pages 3544–3552. Association for Computational Linguistics (ACL).

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263. ACL.

Zeqiu Wu, Michel Galley, Chris Brockett, Yizhe Zhang, Xiang Gao, Chris Quirk, Rik Koncel-Kedziorski, Jianfeng Gao, Hannaneh Hajishirzi, Mari Ostendorf, et al. 2021. A controllable model of grounded response generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 14085–14093.

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.

Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. 2020. Masking as an efficient alternative to finetuning for pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2226–2241.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018. A dataset for document grounded conversations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 708–713.

A   Example Appendix

A.1   Training Details

We use the pre-trained GPT checkpoint trained by Megatron-LM (Shoeybi et al., 2019). When number of samples is less than 5000, we set batch size as 8 and otherwise we set it as 64. We set learning rate as 1e-4 for Xsum dataset when running Adapter (AP) tuning or Prefix tuning (PF). We set it as 3e-4 for WoW dataset. For finetuning over Xsum dataset, we use the learning rate of 3e-5 for 357m and 1e-5 for 1.3b and 8.3b model. When finetuning WoW dataset, we set it for 8e-6 for all model sizes. Xsum dataset is trained for 10 epochs and WoW dataset is trained for 5 epochs. We used ROUGE-L score at the validation set to select the best model for Xsum and used PPL at the seen validation set for WoW. To AP tune 530b MT-NLG model, we set the learning rate as 3e-5 and the batch size as 32.

The prefix length is fixed as 100 and hidden dimension as 800 for PF experiments (Li and Liang, 2021). For AP, the hidden size was set to 50 to achieve a similar extra number of parameters for inference. We summarize the extra task specific parameters introduced by PF and AP in Table 9. Note that we don't intend to do extensive hyperparameter search for all the combinations for model size, sample size and tuning methods. We instead would like to draw conclusions that can generalize across model size, sample size and tasks. We used NVIDIA V100 and A100 GPUs to run all experiments.

Model	Method	Parameter
357m	P-tuning	72k
357m	PT	154k
357m	PF	5m
357m	AP	5m
1.3b	PF	10m
1.3b	AP	10m
8.3b	PF	33m
8.3b	AP	33m

Table 9: Extra parameters for different methods

For summarization, we simply give the article as the input and the summary as the output. For dialogue, We formulate the input with the following template: "{TOPIC}$\backslash t${dialogue_history}$\backslash t$Knowledge: {knowledge}$\backslash t \backslash t$B:". For dialogue history, we add A: and B: in front of each utterance to distinguish different speakers. $\backslash t$ is used to separate different dialogue turns. We use beam search for the decoding step and we set beam size as 5 for all settings.

A.2   More Related Work

More work about faithfulness include summarization (Cao et al., 2018; Dong et al., 2020; Cao and Wang, 2021; Huang et al., 2021), dialogue generations (Rashkin et al., 2021; Shuster et al., 2021; Dziri et al., 2021; Wu et al., 2021), data-to-text (Wiseman et al., 2017; Nie et al., 2019; Liu et al., 2021a; Rebuffel et al., 2022), and translation (Lee et al., 2019; Wang and Sennrich, 2020). However, still relatively little is known about faithfulness/hallucination problem. Pagnoni et al. conduct a good analysis of error types.

Other parameter efficient learning methods includes PPT (Gu et al., 2022), masking (Zhao et al., 2020) with application in multitask learning (Stickland and Murray, 2019; Mahabadi et al., 2021), transfer learning (Pfeiffer et al., 2021; Su et al., 2022; Vu et al., 2022), improving robustness (Han et al., 2021), low resources settings (He et al., 2021b)

A.3   Additional Results

We present detailed ROUGE scores for Xsum in the following tables.

Model	Samples	FT			PF (5m)			AP (5m)		
		R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
357m	1k	28.52	8.07	21.97	30.75	10.19	24.05	31.51	10.55	24.47
357m	5k	32.75	10.80	25.21	32.89	11.56	25.73	32.91	11.45	25.62
357m	10k	34.58	12.54	27.09	33.25	11.76	25.94	33.66	12.05	26.34
357m	50k	38.06	15.69	30.28	34.52	12.73	27.06	36.20	13.90	28.53
357m	100k	39.56	17.13	31.73	34.90	12.93	27.37	37.37	14.96	29.49
357m	200k	41.59	19.21	33.77	35.14	13.26	27.68	37.67	15.36	30.06

Model	Samples	FT			PF (10m)			AP (10m)		
		R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
1.3b	1k	32.73	11.01	25.36	33.61	12.11	26.41	33.94	12.17	26.40
1.3b	5k	34.90	12.49	26.96	36.34	14.34	28.71	36.65	14.61	28.98
1.3b	10k	38.00	15.49	30.28	37.22	15.10	29.57	37.56	15.39	29.83
1.3b	50k	40.92	18.19	32.89	38.60	16.38	30.81	40.09	17.47	32.08
1.3b	100k	42.27	19.48	34.21	39.74	17.21	31.85	41.25	18.57	33.26
1.3b	200k	43.95	21.06	35.75	40.38	17.75	32.37	42.35	19.55	34.22

Model	Samples	FT			PF (33m)			AP (33m)		
		R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
8.3b	1k	37.23	14.87	29.45	37.33	16.17	29.82	39.32	17.28	31.23
8.3b	5k	39.75	16.80	31.58	40.76	18.36	32.70	40.48	17.73	32.21
8.3b	10k	40.73	17.76	32.68	41.78	19.19	33.60	41.33	18.60	33.07
8.3b	50k	43.39	20.21	35.02	43.82	20.99	35.62	43.48	20.35	34.99
8.3b	100k	44.55	21.45	36.13	44.80	21.94	36.45	44.26	21.05	35.80
8.3b	200k	46.10	23.14	37.76	45.51	22.64	37.14	45.24	22.07	36.73

Table 10: Full Rouge score for different model and sample size settings.

Model	Samples	PF (5m)			AP (5m)		
		R-1	R-2	R-L	R-1	R-2	R-L
1.3b	1k	33.78	12.21	26.47	34.35	12.66,	26.83
1.3b	5k	36.38	14.40	28.79	36.41	14.41	28.83
1.3b	10k	37.14	15.08	29.49	37.41	15.28	29.71
1.3b	50k	38.89	16.56	31.06	39.82	17.21	31.86
1.3b	100k	39.78	17.44	32.01	40.80	18.15	32.83
1.3b	200k	40.62	18.07	32.76	41.48	18.97	33.73

Model	Samples	PF (10m)			AP (10m)		
		R-1	R-2	R-L	R-1	R-2	R-L
1.3b	1k	33.61	12.11	26.41	33.94	12.17	26.40
1.3b	5k	36.34	14.34	28.71	36.65	14.61	28.98
1.3b	10k	37.22	15.10	29.57	37.56	15.39	29.83
1.3b	50k	38.60	16.38	30.81	40.09	17.47	32.08
1.3b	100k	39.74	17.21	31.85	41.25	18.57	33.26
1.3b	200k	40.38	17.75	32.37	42.35	19.55	34.22

Model	Samples	PF (5m)			AP (5m)		
		R-1	R-2	R-L	R-1	R-2	R-L
8.3b	5k	40.44	17.93	32.28	41.48	19.04	33.35
8.3b	10k	41.79	19.09	33.45	42.33	19.72	34.22
8.3b	50k	43.62	20.91	35.41	44.39	21.54	36.17
8.3b	100k	44.74	21.79	36.39	45.14,	22.28	36.80
8.3b	200k	45.60	22.68	37.21	46.04	23.16	37.71

Model	Samples	PF (33m)			AP (33m)		
		R-1	R-2	R-L	R-1	R-2	R-L
8.3b	5k	40.76	18.36	32.70	40.48	17.73	32.21
8.3b	10k	41.78	19.19	33.60	41.33	18.60	33.07
8.3b	50k	43.82	20.99	35.62	43.48	20.35	34.99
8.3b	100k	44.80	21.94	36.45	44.26	21.05	35.80
8.3b	200k	45.51	22.64	37.14	45.24	22.07	36.73

Table 11: Adding extra parameters are not always helpful. It happens across different training sample sizes.