ADEPT: Adapter-based Efficient Prompt Tuning Approach for Language Models

Aditya Shah, Surendrabikram Thapa, Aneesh Jain, Lifu Huang
Department of Computer Science, Virginia Tech
{aditya31, surendrabikram, aneeshj, lifuh}@vt.edu

Abstract

Fine-tuning large pre-trained models for downstream tasks can be really expensive. In the past, researchers have proposed various alternatives like adapter and prompt-based methods for tuning these large language models using minimal parameters. However, applying prompt-tuning for smaller language models has not been effective so far and not much work is done in pushing forward soft prompting for these smaller models. To improve the training efficiency of the language models and reduce the size of tuned parameters, we propose a novel Adapter-based Efficient Prompt Tuning approach (ADEPT). In this paper, we show that tuning the parameters of soft prompts with adapter modules while keeping the rest of the model frozen can be a promising method to optimize smaller language models for downstream tasks. Our method achieves up to *98% performance of full fine-tuning while using only 0.02% of total model parameters*.

1 Introduction

With the rapid advancement in computational facilities and the research in the field of Natural Language Processing (NLP), pre-trained language models (Peters et al., 2018; Conneau et al., 2018; Devlin et al., 2019; Yang et al., 2019; Raffel et al., 2020; Qiu et al., 2020; Xue et al., 2021; Doddapaneni et al., 2021) have been used widely in various tasks. These models use different statistical and probabilistic methods to decide the likelihood of a given sequence of words occurring in a sentence. To efficiently use the language models, researchers fine-tune these pre-trained language models on downstream tasks. With the conventional practices of fine-tuning the models, new parameters are generally introduced for every downstream task. However, this approach of fine-tuning language models becomes difficult especially when there are a lot of trainable parameters. With language models becoming larger and larger (Brown et al., 2020), we can often anticipate challenges related to maintaining multiple copies of model parameters for inference, training time, and lack of necessary computing power. The concept of prompt-tuning was introduced to improve parameter efficiency in the downstream tasks. In prompt tuning, there's no need for new parameters as we convert our problem into a language modeling task. This method can be promising, especially when there are very few training examples for e.g. few shot learning (Gao et al., 2021), where the standard fine-tuning would not be efficient.

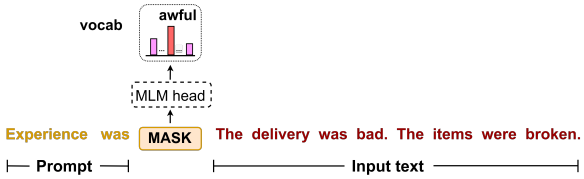


Figure 1: Prompt-based tuning using discrete prompt. The prompt "*Experience was*" with a `[MASK]` is prepended to the input text.

A *prompt* is usually a sequence of words or parameters that are appended or prepended to the input so that the given downstream task can be constructed as a language modeling problem (Liu et al., 2021a). An example is shown in Figure 1. In order to classify the sentiment of a given text, like an amazon product review "*The delivery was bad. The items were broken.*", we can prepend a prompt, such as "*Experience was*" or "*It was*", with a `[MASK]` to the sentence and anticipate the language model to predict a negative adjective such as "*awful*" for the `[MASK]` position.

As shown by Lester et al. (2021); Kim et al. (2021), soft prompt-tuning for smaller language models do not perform well when compared to traditional fine-tuning. This approach only works for significantly larger models ($\text{BERT}_{\text{large}}$, $\text{RoBERTa}_{\text{large}}$, etc). In this paper, we propose a novel approach to leverage soft prompt tuning for smaller language models. We insert adapter modules in the language model, then jointly fine-tune the parameters of this adapter module along with soft prompts while keeping the rest of the model frozen. Through empirical results on 3 benchmark datasets from SuperGLUE (Wang et al., 2019) and 4 text classification datasets, we demonstrate the effectiveness of our proposed approach for these

Proceedings of The Fourth Workshop on Simple and Efficient Natural Language Processing (SustaiNLP), pages 121–128

smaller LM models (RoBERTa (Liu et al., 2019) and BERT (Devlin et al., 2019)). Our method optimizes only 0.02% of total model parameters during training and yet achieves better performance than other tuning strategies while being competitive to fine-tuning.

Our main contributions can be summarised as follows:

- a new Adapter-based Efficient Prompt Tuning (ADEPT) approach to leverage soft prompts for smaller language models - *"roberta-base"* and *"bert-base-cased"*.
- analyze the effectiveness of our approach with respect to other soft prompt-tuning and fine-tuning methods.
- an ablation study to investigate the importance of the number of prompt tokens and adapter hidden size.

2 Related Work

The effectiveness of prompt tuning was demonstrated by (Brown et al., 2020) where the authors showed that GPT-3 model could handle wide variety of tasks using only a few training examples. The use of prompts was first proposed by (Radford and Narasimhan, 2018). The authors showed that language models can perform well in few-shot and zero-shot settings through these natural language prompts. More recently, Jiang et al. (2020) proposed an approach to automatically discover better prompts in order to improve the factual knowledge retrieval from these language models. Moreover, Schick and Schütze (2021) introduced Pattern Exploiting Training (PET) which uses cloze-style phrases and achieves state-of-the-art performance on few supervised and semi-supervised tasks - classification on Yelp Reviews, AG's News, Yahoo Questions (Zhang et al., 2015) and MNLI (Williams et al., 2018). This work was further improved by (Tam et al., 2021) for few-shot natural language understanding without using any unlabeled data. In all of these approaches, prompts were manually designed in the form of discrete tokens. Thus, in such scenarios, it is important to design appropriate prompts based on different downstream tasks. The importance of prompt engineering and the complete paradigm of prompt tuning is summarized in Liu et al. (2021a).

In contrast to discrete prompts (Shin et al., 2020; Hambardzumyan et al., 2021; Gao et al., 2021; Reynolds and McDonell, 2021), soft prompts are randomly initialized vectors that are prepended or appended to the input text. The parameters of the entire language model are fixed and only the prompt parameters are fine-tuned. Liu et al. (2021b) showed that automatically searching better prompts in the continuous space gives competitive performance for natural language understanding. Soft prompts were initially proposed by Zhong et al. (2021) where OptiPrompt was proposed and outperformed discrete prompts on knowledge probing tasks. Li and Liang (2021) and Qin and Eisner (2021) used a similar idea for generation tasks where they prepended task-specific prompts in the input text and achieved comparable performance as the original model fine-tuning. Han et al. (2021) proposed prompt-tuning with rules (PTR) which significantly outperformed state-of-the-art baselines for relation classification tasks. The effectiveness of soft prompt-tuning was further leveraged by Lester et al. (2021) where they applied soft prompts on the T5 model and achieved a good performance on the SuperGLUE benchmark. Furthermore, Su et al. (2021); Vu et al. (2022) studied the transferability of prompt tuning across different tasks and models. They showed that soft prompts can be transferred to similar tasks without training and can be a good initialization for the underlying language model.

3 Approach

We propose a novel adapter-based prompt-tuning architecture for downstream classification tasks. Figure 2 shows an overview of the model. We use the pre-trained BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) from Hugging Face as the encoder. For soft prompt-tuning, we append soft prompt embedding in the input text using the same approach as Lester et al. (2021). Given an input text $X = [x_1, ..., x_n]$, where x_i is the $i-$th token in the text and n is the length of the sequence, we append prompt tokens $[p_1, p_2, ... p_m]$ where m is the number of prompt tokens. We initialize these prompt tokens with embeddings obtained from random words in the vocabulary and update them during training, which shows better performance on downstream tasks than initializing them with random vectors. We found that initializing prompt tokens with random word embeddings instead of random vectors helps the model converge better on downstream tasks. The resulting input to the

model becomes: $[p_1, ..., p_m, x_1, ..., x_n]$. We do not use any separator token between prompt vectors and tokens.

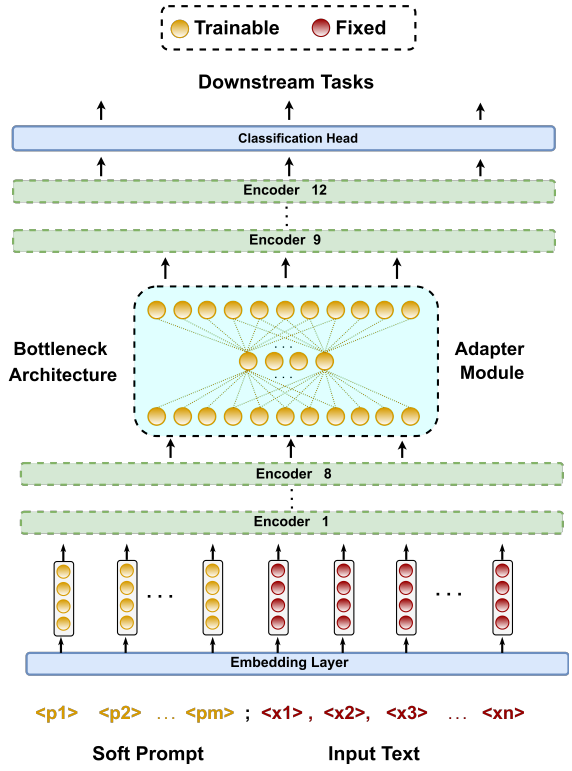


Figure 2: Overview of the ADEPT model. The adapter module is inserted between encoder layer 8 and layer 9. Parameters of prompt embeddings and this adapter module are tuned while the rest of the complete model and input text embeddings are frozen. ADEPT approach using RoBERTa/BERT tunes only $28K$ parameters (0.02% of $123M$ total parameters)

Inspired by the work of Bapna and Firat (2019); Houlsby et al. (2019); Pfeiffer et al. (2021); Rücklé et al. (2021); He et al. (2021), we insert adapter modules in the form of Multi-Layer Perceptron (MLP) between the encoder layers of the base transformer model. To reduce the number of parameters, we choose a bottleneck architecture for the adapter module. In a bottleneck architecture, the hidden layer also called as bottleneck size is much smaller than the input layer. If d is the input size (number of neurons) of the MLP layer and b is the bottleneck size (adapter hidden size), then the bottleneck architecture would require $2 \cdot d \cdot b + d + b$ parameters . When $b << d$, the total parameters are greatly reduced compared to a general MLP layer with $d \cdot d + d$ parameters. If the embedding dimension size is e and the prompt length is m, then the trainable parameters for prompt embeddings is m . e. So the total trainable parameters for the ADEPT model are just $[2 \cdot d \cdot b + (d + b)] + m \cdot e$. We use one adapter module with hidden size $b = 8$ and prompt length $m = 20$ in our implementation which is inserted between the 8^{th} and 9^{th} encoder layer [1]. For RoBERTa and BERT models, we have $d = e = 768$. So, total trainable parameters for the ADEPT model is only $28k$ while in standard fine-tuning, we update all the parameters ($123M$ parameters for *roberta-base* model.)

4 Experimental Setup

We conduct experiments on four classification datasets - *IMDB* (Maas et al., 2011), *AG's News* (Zhang et al., 2015) , *Yahoo Answers* (Zhang et al., 2015), Yelp - 5 (Zhang et al., 2015) and three datasets from SuperGLUE benchmark - Boolq (Clark et al., 2019), CommitmentBank (de Marneffe et al., 2019), and Recognizing Textual Entailment (RTE) . More details on data statistics and implementation details are described in Appendix A

To show the efficiency of our adapter-based prompt tuning approach, we compare it with several other training paradigms:

- **Prompt-tuning (PT):** fine-tune soft prompt embeddings while keeping the entire model frozen. The randomly initialized parameters of the classification head are also fixed.
- **Head-tuning (HT):** only fine-tune the classification head. The base transformer model is frozen.
- **Prompt + Head-tuning (PHT):** fine-tune soft prompt embeddings along with the parameters of the classification head. The base transformer model is frozen.
- **Fine-tuning (FT):** traditional fine-tuning where the entire model along with the classification head is trained.

[1] We experimented with different positions for the adapter module: between layer 4 and 5; between layer 6 and 7; between layer 8 and 9; between layer 10 and 11. We achieved the best results when the adapter module was inserted between layer 8 and layer 9. It stands out to the reason that the lower layers of BERT account for sentence and coarse linguistic structure while the higher layers are more domain-specific and help to learn task-specific parameters (Rogers et al., 2020). Attaching an adapter module between higher layers helps the adapter module parameters to converge better to downstream tasks

Model	Method	IMDB	AG's News	Yahoo	Yelp-5	BoolQ	CB	RTE
BERT	Prompt (PT)	0.91	0.86	0.66	0.57	0.62	0.66	0.47
	Head (HT)	0.89	0.88	0.67	0.60	0.61	0.72	0.53
	Prompt + Head (PHT)	0.92	0.92	0.70	0.63	0.67	0.71	0.54
	Fine-tune (FT)	0.94	0.94	0.72	0.69	0.73	0.83	0.61
	Adapter + Prompt (ADEPT)	0.92	0.93	0.71	0.67	0.69	0.80	0.58
RoBERTa	Prompt (PT)	0.90	0.87	0.65	0.59	0.62	0.64	0.50
	Head (HT)	0.91	0.90	0.68	0.61	0.63	0.70	0.54
	Prompt + Head (PHT)	0.94	0.93	0.72	0.66	0.64	0.71	0.55
	Fine-tune (FT)	0.96	0.95	0.73	0.71	0.80	0.86	0.71
	Adapter + Prompt (ADEPT)	0.95	0.94	0.72	0.68	0.71	0.77	0.60

Table 1: Performance on all evaluation tasks. Each experiment is run for 3 trials and the average result is reported. For text classification tasks (IMDB, AG's News, Yahoo, Yelp-5), we report the test F-score; for Boolq, CommitmentBank (CB), and Recognizing Textual Entailment (RTE), we report the test accuracy. ADEPT approach results in better performance than Head-tuning (HT) and Prompt + Head-tuning (PHT) while using significantly lower parameters and is competitive to standard fine-tuning

- **Adapter + Prompt-tuning (ADEPT):** fine-tune soft prompt embeddings and adapter module parameters that are inserted in the encoder layer. The rest of the model and the randomly initialized parameters of the classification head are kept fixed.

5 Results

Table 1 shows the F-score and accuracy on all evaluation tasks under different training settings. Table 2 compares the model parameters and training metrics. We can see that ADEPT outperforms all the other methods which just tune the prompt or head layers using significantly lower parameters. By just tuning 0.02% parameters, ADEPT shows comparable performance as the method that fine-tunes all the parameters, demonstrating its significance in improving the training efficiency. It is interesting to see that although Head-tuning (HT) and Prompt + Head-tuning (PHT) methods have larger parameters to be optimized, the ADEPT method still achieves better results on the SuperGLUE dataset and requires lesser training time compared to the standard fine-tuning approach. We also observe that standard prompt-tuning requires about 2.5 times more steps to converge compared to fine-tuning approach and does not perform well for multi-class classification. A similar result was reported by Lester et al. (2021), where the authors claim that soft prompt based-tuning does not perform well for smaller language models.

Overall, attaching adapter modules between the encoder layers helps to retain the knowledge from pre-trained LMs and efficiently learn the required parameters to further improve the performance on downstream tasks. This helps the language model to better adapt to downstream tasks using minimal parameters. It achieves comparable performance as standard fine-tuning while being highly parameter efficient and requiring much lower training time. All these demonstrate the effectiveness of our proposed approach in improving the training and resource efficiency of language models.

Method	# params	% params	time	Convergence
PT	*13K*	0.01%	0.9 *t*	2.5 *x* steps
HT	*600K*	0.48%	0.5 *t*	2 *x* steps
PHT	*620K*	0.49%	0.8 *t*	2 *x* steps
FT	*123M*	100%	*t*	*x* steps
ADEPT	*28K*	0.02%	0.7 *t*	1.5 *x* steps

Table 2: Model parameters and training metrics. *t* and *x* refer to training time and training steps respectively as required by standard fine-tuning. # params denotes the number of trainable parameters for every method. % params denotes % of parameters to be optimized compared to standard fine-tuning.

6 Conclusion

We proposed a novel adapter-based prompt-tuning approach for fine-tuning language models. Our results demonstrate the effectiveness of the approach on seven benchmark datasets. ADEPT achieves a significant performance boost over PT, HT, and PHT approaches and is comparable to the standard fine-tuning while using only 0.02% of the model parameters. Since adapter modules learn the required parameters for various NLU tasks, they help in retaining the knowledge of the pre-trained LM model when the encoder is frozen. Our work aims to facilitate the further research direction of using adapter-based prompt methods for tuning LMs.

Limitations

Due to the limited resources, we could not experiment with this approach for larger language models such as *roberta-large* and *bert-large*. It would be interesting to investigate the performance of ADEPT with larger LMs.

In addition, we only evaluate the ADEPT approach on seven downstream tasks. It would also be interesting to test it on more broad natural language processing tasks, such as information extraction, natural language generation, question answering, and so on.

Broader Impact

As discussed earlier, fine-tuning large pre-trained models for downstream tasks can be really expensive. The ADEPT approach can help AI practitioners to assess the abilities of LM without using a lot of resources. This approach of prompt tuning can also help smaller end users to take advantage of harnessing the power of LM with the minimal resources they have. It can be used by social scientists, Non-profit organizations, etc. to create a positive impact in society in spite of limited computing resources.

Reproducibility

The code and resources for this work are available at our GitHub repository[2]. The code for training, testing, and producing plots are made available. The details on the model and hyperparameters are given in Appendix A.2. A brief introduction to the dataset used in this paper is given in Appendix A.1. Our implementation approach specific to all the dataset used are also explained.

References

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse.

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

Sumanth Doddapaneni, Gowtham Ramesh, Mitesh M. Khapra, Anoop Kunchukuttan, and Pratyush Kumar. 2021. A primer on pretrained multilingual language models.

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

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. WARP: Word-level Adversarial ReProgramming. 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 4921–4933, Online. Association for Computational Linguistics.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification.

[2]https://github.com/Aditya-shahh/ADEPT

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

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

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know?

Boseop Kim, HyoungSeok Kim, Sang-Woo Lee, Gichang Lee, Donghyun Kwak, Jeon Dong Hyeon, Sunghyun Park, Sungju Kim, Seonhoon Kim, Dongpil Seo, Heungsub Lee, Minyoung Jeong, Sungjae Lee, Minsub Kim, Suk Hyun Ko, Seokhun Kim, Taeyong Park, Jinuk Kim, Soyoung Kang, Na-Hyeon Ryu, Kang Min Yoo, Minsuk Chang, Soobin Suh, Sookyo In, Jinseong Park, Kyungduk Kim, Hiun Kim, Jisu Jeong, Yong Goo Yeo, Donghoon Ham, Dongju Park, Min Young Lee, Jaewook Kang, Inho Kang, Jung-Woo Ha, Woomyoung Park, and Nako Sung. 2021. What changes can large-scale language models bring? intensive study on HyperCLOVA: Billions-scale Korean generative pretrained transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3405–3424, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

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

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. GPT understands, too. *CoRR*, abs/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.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

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

Jonas Pfeiffer, 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, Online. Association for Computational Linguistics.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212, Online. Association for Computational Linguistics.

XiPeng Qiu, TianXiang Sun, YiGe Xu, YunFan Shao, Ning Dai, and XuanJing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

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

Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2021. AdapterDrop: On the efficiency of adapters in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural*

Language Processing, pages 7930–7946, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze questions for few shot text classification and natural language inference.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Zhiyuan Liu, Peng Li, Juanzi Li, Lei Hou, Maosong Sun, and Jie Zhou. 2021. On transferability of prompt tuning for natural language understanding.

Derek Tam, Rakesh R Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training.

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, Dublin, Ireland. Association for Computational Linguistics.

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

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

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *CoRR*, abs/1509.01626.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [mask]: Learning vs. learning to recall.

A Appendix

A.1 Dataset

The data statistics along with number of examples and labels is shown in Table 3.

Dataset	# train	# val	# test	# labels
IMDB	40,000	5,000	5,000	2
AG's News	102,000	18,000	7,600	4
Yahoo	1,300,000	60,000	100,000	10
Yelp-5	520,000	130,000	50,000	5
Boolq	9,427	3,270	3,245	3
CB	250	57	250	3
RTE	2,500	278	300	2

Table 3: Data statistics

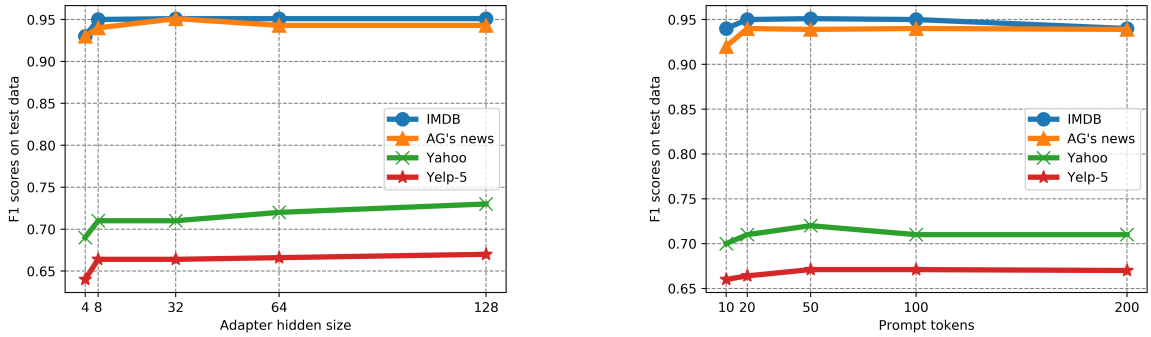
IMDB: It is a binary sentiment analysis dataset that has a movie review and its respective label. The dataset consist of 2 labels - *"positive"* and *"negative"*.

AG's News: A news classification dataset where every example consists of a headline h, a text body b, and a label associated with it. The dataset consist of 4 labels - *"World"*, *"Sports"*, *"Business"*, and *"Science/Tech"*. For our implementation, we concatenate the headline and body - $[h : b]$ and use the concatenated text for the prediction.

Yahoo Answers: A topic classification dataset that consists of a question q, an answer a, and a label associated with it. The dataset consists of 10 labels - *"Society"*, *"Science"*, *"Health"*, *"Education"*, *"Computer"*, *"Sports"*, *"Business"*, *"Entertainment"*, *"Relationship"*, and *"Politics"*. For our implementation, we concatenate the question and answer - $[q : a]$ and use the concatenated text for the prediction.

Yelp-5 A review classification dataset where every example consists of a review and a label associated with it. The dataset consists of 5 labels in the form of numbers - (1 to 5).

Boolq Boolq is a question answering dataset for yes/no questions from the SuperGLUE benchmark. Each example is a triplet of (question, passage, and



(a) Varying adapter size

(b) Varying # of prompt tokens

Figure 3: Test F-scores on the classification datasets using RoBERTa with ADEPT model In (a), number of prompt tokens is kept fixed $(m) = 20$ and adapter hidden size is varied. In (b), the adapter hidden size is kept fixed $(d) = 8$ and number of prompt tokens is varied.

answer). For our implementation, we concatenate the question and passage - *[q : p]* and use the concatenated text for the prediction. The dataset consist of 2 labels - *"yes"* and *"no"*

CB CommitmentBank (CB) is a three-class textual entailment dataset from the SuperGLUE benchmark. Each example is a triplet of (premise, hypothesis, and label). For our implementation, we concatenate the premise and hypothesis - *[p : h]* and use the concatenated text for the prediction. The dataset consist of 3 labels - *"contradiction"*, *"neutral"*, and *"entailment"*.

RTE Recognizing Textual Entailment (RTE) is a binary textual entailment dataset from the SuperGLUE benchmark. Each example is a triplet of (premise, hypothesis, and label). For our implementation, we concatenate the premise and hypothesis - *[p : h]* and use the concatenated text for the prediction. The dataset consist of 2 labels - *"entailment"* and *"not_entailment"*.

A.2 Implementation details

We use *roberta-base* and *bert-base-cased* from Hugging Face[3]. For our experiment, the adapter module is inserted between the 8^{th} encoder layer and 9^{th} encoder layer of both models. We run each experiment for 3 trials and the average result is reported. The learning rate is $8e-4$ for ADEPT, prompt-tuning (PT), prompt-tuning + head (PHT), head-tuning (HT), and $2e-5$ for fine-tuning method. The batch size is 16 for all the methods. We train fine-tuning method for 10 epochs, the ADEPT model for 15 epochs, prompt-tuning + head and head-tuning for 20 epochs, and prompt-tuning for 25 epochs. We use AdamW optimizer and linear scheduler with 6% warmup steps for all the methods. For the ADEPT model, we use an adapter hidden size of 8 and the number of prompt tokens is 20 for ADEPT, PT, and PHT methods.

A.3 Ablation Study

We analyze RoBERTa with the ADEPT model to show the impact of adapter hidden size and number of prompt tokens. We conduct two sets of experiments Figure 3a - for different adapter sizes and Figure 3b - for different prompt tokens.

Adapter Hidden Size: We train the ADEPT model for varying adapter sizes - (4, 8, 32, 64, 128), and the number of prompt tokens is kept as 20. As Figure 3a shows, the F-score is slightly improved as we increase the adapter size. Simpler data like IMDB do not benefit much from the increase in adapter size. Increasing adapter size beyond 32 brings 2 to 3% improvement for Yahoo and Yelp-5 dataset

Number of Prompt Tokens: We train the ADEPT model by varying the number of prompt tokens - (10, 20, 50, 100, 200) and keep the adapter hidden size fixed as 8. When the number of prompt tokens is beyond 50, the performance is decreased, indicating that adding trainable parameters in the input does not help much beyond a certain point.

[3]https://huggingface.co/