On the Domain Robustness with Prompt & Prefix Tuning

Colin Wang[*†] **Lechuan Wang**[*] **Yutong Luo**
Halıcıoğlu Data Science Institute
University of California, San Diego
{ziw029, l6wang, y5luo}@ucsd.edu

Abstract

Prompt tuning and prefix tuning are two effective mechanisms to leverage frozen language models to perform downstream tasks. Robustness reflects models' resilience of output under a change or noise in the input. In this paper, we analyze the robustness of natural language models using various tuning methods with respect to a domain shift (i.e. training on a domain but evaluating on out-of-domain data). We apply both prompt tuning and prefix tuning on T5 models for reading comprehension (i.e. question-answering) and GPT-2 models for table-to-text generation. Our results demonstrate significant divergence in domain robustness patterns given two similar prompt tuning methods under relatively fair experimental settings. We further propose future research directions to explore and validate the causes of such differences.

1 Introduction

NLP models have recently achieved outstanding performances and are thus gained prevalent applications in real world (Bahdanau et al., 2015; Hu and Li, 2021; See et al., 2017). With this popularity, it is important to make sure these models could adapt well in the dynamic circumstances. More specifically, robustness with respect to domain shifts is supposed to be considered when developing models (Ramponi and Plank, 2020; Wang et al., 2019a). Because the same large pre-trained language models are often applied to different tasks or fields. It would be inefficient and impractical if we train the model with corresponding inputs every time we apply them to a different domain. We want large models can be easily reused and adapted to various tasks and domains. Improvement on models to ensure they are robust against change of inputs modality and domain has been a hot topic for study (Wei et al., 2022; Ye et al., 2021).

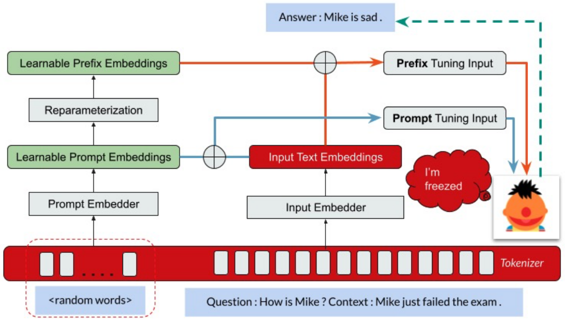

Figure 1: Architectural difference between prompt tuning and prefix tuning. In prompt tuning, the initial prompt embedding is directly derived from randomly initialized words from vocabulary. The parameters in the prompt embedding are directly updated as we fine-tune the model for downstream tasks. In prefix tuning, in addition to having the prompt embedding, the embedding is further reparamterized and activated to form the prefix embedding. In back-propagation, the pre-trained model itself is freezed, only allowing the parameters from the prompt/prefix embeddings to be updated.

With the advance of NLP, a wide range of mechanisms have been developed to adjust large pre-trained language models to downstream tasks. To avoid the update and storage of language model parameters, Li and Liang (2021) developed prefix tuning, which freezes the parameters of language model, and only optimizes the small continuous task-specific vector (i.e. the prefix). They apply prefix tuning on GPT-2 models (Radford et al., 2019), and find great model performances under different data settings.

Prompt tuning (Lester et al., 2021) is proposed as a further simplification of prefix tuning. Similar to prefix tuning, the pretrained language model is kept freezed, but this time, prompt tuning directly applies learnable soft embedding to being concatenated with the input embedding. With the

[*] Equal technical contributions.

[†] Work done in third-year. Colin will continue the final year of undergraduate studies after this capstone project.

[‡] Code available at https://github.com/zwcolin/Domain-Robustness-Prompt-Tuning

end-to-end employment of prompt tokens, prompt tuning achieves outperforming results and efficient model reuse on T5 models (Raffel et al., 2020).

2 Related Works

Language Model fine-tuning is the core procedure to adapt a pre-trained model to a downstream task. The essence of fine-tuning is by updating the model parameters from learning the input-output pair of a given task and some data. Many recent works have been focusing on reducing the number of parameters needed fot a language model to adapt to a downstream task (thus subsequently reduce computational and storage needs as well). Houlsby et al. (2019) proposed a adapter module that achieved superior performance by integrating the transformer layer with such adapter modules, where fewer updated parameters (compared to model-tuning on top layers) lead to greater gains. Hambardzumyan et al. (2021) proposed an adversarial approach to learn task-specific word embedding, which concatenate with the input text to instruct the model to generate desired outputs. This work is different from Li and Liang (2021) and Lester et al. (2021) that its methodology limits the model to produce only 1-token output, which is mostly useful for classification tasks. Gao et al. (2021) introduced prompting for few shot-learning, but their method is highly dependent on manual prompt engineering compared our main objective methods, which learn the soft prompt based on the data. Our works will focus on prompt tuning and prefix tuning as they are most flexible in terms of model and task compatibility.

Domain adaptation has many objectives, but most of them aim to enable the model to consistently produce desirable performance for the same task over various domains when the model is exposed to certain domains. Wang et al. (2019b) and Cao et al. (2020) used adversarial training paradigm to not only fine-tune a language model for a downstream task, but also asks the model to classify the domain that this input belongs to, making it domain invarient. Xu et al. (2019) proposed multiple auxiliary loss terms to prevent the model from catastrophic forgetting on its original domain when it's fine-tuned on a different domain. Zhang et al. (2020) demonstrated that augmenting pretraining dataset with a certain low-resource domain would greatly improve the model performance later when it's fine-tuned on the same domain. Finally, Talmor and Berant (2019) showed that by combining training data from multiple domains, the model generalizes better to other domains, and further, by a two-step fine-tuning from a large out-of-domain dataset to a small in-domain dataset under the task and input modality, the model needs less data to achieve a good performance on the in-domain dataset compared when it is directly fine-tuned on the in-domain dataset. Our main focus will be on exploring prompt & prefix tuning's ability for zero-shot domain adaptation due to fewer parameter needs. We hypothesize that because these prompt tuning methods require smaller amount of parameters, the model has to learn what is commonly existing (i.e. the same task methodology) in data coming from different domains, rather than domains or data themselves.

3 Experimental Setup

3.1 Datasets and Metrics

For GPT-2 model, we investigate the domain robustness on Table-to-Text generations. We train the model on WebNLG (Colin et al., 2016), and test on DART (Bosc et al., 2016). DART is more complex and has larger size than WebNLG. DART is open-domain while WebNLG has only 14 domains. We evaluate the performance using BLEU (Papineni et al., 2002) score, which is reported by the official evaluation script for WebNLG and DART. We will also include METEOR (Banerjee and Lavie, 2005) and TER (Snover et al., 2006) score, which measures the translation accuracy.

The WebNLG (Colin et al., 2016) corpus comprises of 25,298 (data, text) pairs and 9,674 sets of triplets (subject, property, object) describing facts (entities and relations between them) and the corresponding facts in form of natural language texts. The test set is split into two parts: on one hand, it contains DBpedia categories that were seen in the training data; and on the other hand, it consists of inputs from 5 unseen categories.

DART (Bosc et al., 2016) is a large dataset for open-domain structured data record to text generation. It has a similar input format to WebNLG but is richer and more diverse than WebNLG. DART consists of 82,191 examples across different domains with hierarchical inputs based on a tree ontology that transforms a flat table into a tree structure.

For T5 models, we investigate the domain robustness on question-answering tasks. In our experiments, We train our model on the SQuAD (Rajpurkar et al., 2016) dataset, and test on the DuoRC (Saha et al., 2018) dataset. The evaluation metric for T5 is EM/F1 score, which is derived from the script provided by the MRQA challenge by Fisch et al. (2019).

SQuAD (Rajpurkar et al., 2016) is a reading comprehension dataset, containing 107,785 question-answer pairs. Questions in this dataset are posed by crowdworkers from Wikipedia articles, and the answer to every question is a segment of text from the corresponding reading passage, meaning the system will select the answer from all possible spans. Even though span-based answers are more constrained, SQuAD dataset still provides us with diverse questions and answer types.

DuoRC (Saha et al., 2018) is another dataset for reading comprehension dataset. DuoRC contains 186,089 (question,answer) pairs generated from a collection of 7680 pairs of movie plots. Every pair in the collection reflects two versions of the same movie since they are written by two different groups of crowdworkers. This makes the answers less overlapping, different in levels of plot details and higher requirements for reasoning process.

We also conduct an additional experiment that aims to address the question whether there are certain domains that would make the model generalize better to other domains, and we make full use of MRQA dataset by considering each of its subset as representing a domain.

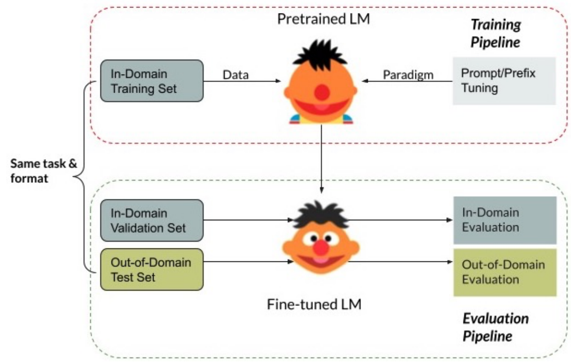

Figure 2: Generalized experimentation setup. To test the domain robustness of a specific prompt/prefix tuning method, we use the same pre-trained model and a method to fine-tune on an in-domain training set. Then, the fine-tuned model is evaluated on in-domain validation set for in-domain performance and on out-of-domain test set for out-of-domain performance. All three datasets share the same task and input formats.

3.2 Methods & Hyperparameters

In our work, we will apply both prompt and prefix tuning on T5 and GPT-2 models. Our experimental design spans two dimensions for each model and tuning method. First, we measure the robustness of tuning with respect to different model sizes, given the same prompt length. Second, we measure the robustness of tuning with respect to different prompt lengths, given the same model size. We train T5 with sizes ranging from small, base and large, and GPT-2 ranging from medium and large. Both models are trained with different token lengths including 1, 5, 10, 20, and 50. The prompts and prefixes' parameters are initialized from vocabulary.

For the T5 model on question-answering, we trained it with AdaFactor (Shazeer and Stern, 2018) with a learning rate of 0.001 and a linear learning rate annealing scheme. In terms of the optimizer, we disabled scaling the parameter and the relative step. We used a clip threshold of 1.0, and we did not have any warm up steps during training. We run 4 epochs through all the training data in our experiments. This applies to both prompt tuning and prefix tuning. We did not perform any hyperparameter search in our experimental setting and the given choices are what most people have been using. We hope that this setting will show us a more natural and realistic performance of the model.

For the GPT-2 model on table-to-text generation with prefix tuning, we followed the optimized parameters provided by Prefix-tuning (Li and Liang, 2021). In particular, we trained it with AdamW optimizer (Loshchilov and Hutter, 2019) and a linear learning rate scheduler according to the HuggingFace default setup. The learning rate is $5 \cdot 10^{-5}$. On the other hand, for prompt tuning, we trained it with the same hyperparameter set that has been used for prompt tuning the T5 model. The reason that we did not use the hyperparameters provided in prefix-tuning for prompt tuning in this experiment is due to its poor convengence.

4 Results

4.1 T5 & Question Answering

Our experimentation results for prompt/prefix tuning on T5 model for question-answer task are provided in Table 1. There are several discoveries that we have found under our experimentation settings:

First, prefix tuning is better in general for smaller models. Under the small-sized T5 model, prefix

Configurations		**In-Domain**				**Out-of-Domain**			
		Prompt		Prefix		Prompt		Prefix	
Size	# Tkns	EM	F1	EM	F1	EM	F1	EM	F1
Small	1	17.86	56.88	75.01	84.1	2.27	25.17	29.51	37.9
	5	21.52	55.61	75.61	**84.32**	2.4	21.48	29.85	38.09
	10	21.97	57.19	75.41	84.24	3.06	23.6	30.05	38.07
	20	27.72	61.08	75.38	84.21	3.53	24.32	30.31	38.18
	50	24.34	60.05	75.36	84.21	3.6	24.76	30.45	**38.34**
Base	1	55.29	79.84	82.54	90.4	30.71	**49.74**	35.04	44.49
	5	47.7	72.44	82.47	**90.42**	18.79	36.13	34.84	44.26
	10	50.09	73.32	82.49	90.29	21.99	39.44	34.98	44.37
	20	55.73	75.95	82.69	90.37	25.98	42.38	34.84	44.14
	50	49.29	74.23	82.33	90.3	16.06	38.11	34.18	43.8
Large	1	55.65	82.01	86.02	**93.21**	49.43	**63.77**	38.24	47.48
	5	49.72	78.89	86.18	93.09	43.84	61.08	38.44	47.45
	10	46.87	78.33	86.15	93.01	46.77	62.11	38.84	47.63
	20	33.67	73.61	86.2	93.17	32.91	56.63	38.44	47.31
	50	40.9	76.35	86.33	93.11	38.97	59.32	38.51	47.28

Table 1: T5 results on question-answering task with prompt & prefix tuning. Here, SQuAD dataset was used as the training set to train the model. In-Domain evaluation metrics are reported based on the validation set of SQuAD dataset, while Out-of-Domain evaluation metrics are reported based on the test set of DuoRC dataset. All data came from the MRQA dataset.

tuning performed better in both in-domain validation dataset and out-of-domain test dataset. Specifically, When the model is fine-tuned with prefix tuning for this task, the best performance under small model is 10-20 points higher in terms of F1 score than is fine-tuned with prompt-tuning.

Second, prompt tuning seems to be superior than prefix tuning as we get larger and larger models. Although prefix tuning has an advantage in F1 score for in-domain data, prompt tuning has a significantly higher F1 score for out-of-domain data, representing a relatively stronger ability to adapt to the new domain. We think this might be caused by the fact that prefix tuning has more parameters than prompt tuning, and when the model also gets larger, this effect is enlarged because the dimension of the prompt or prefix token also gets larger, leading to some overfitting to the training domain for prefix tuning. But this cannot explain the fact that prefix tuning with only 1 token still performs poorly in domain adaptation, and more works need to be done in order to investigate the root cause.

Third, prompt tuning's token choices are model-size agnostic with T5 on question answering. It seems that a token size of 1 always yields the best performance on out-of-domain data. On the other hand, this model-size agnostic pattern is not appearing for prefix-tuning, where there does not seem to have a number of tokens that consistently perform better on out-of-domain data given a certain model size.

Fourth, prompt tuning's performance diverges significantly when having different number of tokens, while prefix tuning's performance keeps consistently over different number of tokens for the same-sized model. This shows that determining a prompt length in prompt tuning is more important than determining a prefix length in prefix tuning.

As an extended part of this experiment, we also attempted to address whether certain domains would make a model generalize to other domains. Previously, Talmor and Berant (2019) has performed similar experiments to show how well a model can generalize to various domains given some particular datasets for training. Their work shows that zero-shot model performance on out-of-domain dataset varies moderately given different training set for fine-tuning the model. Our experiment differs from theirs in the following perspectives: first, we did not use language models that are specially for QA tasks - whereas they DOCQA (Clark and Gardner, 2018) and BERTQA (Devlin et al., 2019) as their models for fine tuning, we directly used a pretrained T5 model for fine tuning;

Evaluation		Training Dataset				
Dataset	Metric	SQuAD	NewsQA	SearchQA	TriviaQA	HotpotQA
SQuAD	EM	52.66	54.9	55.3	54.1	54.4
	F1	78.55	79.24	**79.35**	79.24	79.08
NewsQA	EM	17.8	16.2	18.7	18.9	18.6
	F1	47.64	46.81	47.91	49.19	**49.45**
SearchQA	EM	6.1	6.1	5.5	6	4
	F1	**21.25**	21.24	20.42	21.16	18.9
TriviaQA	EM	27.6	27.8	28.6	27.1	28.8
	F1	55.8	55.88	56.06	56.09	**56.37**
HotpotQA	EM	29.7	29.8	29.8	29.7	31.1
	F1	56.5	56.48	**56.55**	56.49	55.58
BioASQ	EM	34.5	34.6	34.5	34.6	34.5
	F1	55.3	**55.37**	55.32	55.35	55.35
DROP	EM	15.4	15.5	15.4	15.4	15.6
	F1	31.38	31.5	31.39	31.5	**31.54**
RE	EM	44.2	44.5	44.4	44.5	44.5
	F1	73.22	**73.33**	73.21	73.24	73.26
DuoRC	EM	30.7	30.5	30.4	30.6	30.5
	F1	**50.26**	50.19	50.18	50.24	50.17

Table 2: T5 results on question-answering task with prompt tuning. Different datasets for the same task are used to train a model one at a time, and then the model is used to make predictions from out-of-distribution evaluation datasets. We randomly sample 1,000 examples for training, and then make evaluations on 1,000 examples randomly sampled from the corresponding evaluation dataset. To ensure fairness, a seed is used. The model is T5-base with 1 token for prompt tuning, which has been shown to have fast convergence over small amount of data.

second, while they used model fine-tuning to fine tune the model for the training dataset, our method uses prompt tuning, which keeps the pretrained model parameters freezed, while only updating the prompt parameters.

Based on Table 2 from our experimentation setup, we found that data-wise, there does not seem to exist a domain (in the scope of our dataset) that would make the model generalize better to other domains. On the other hand, the prompt trained on any of the domain-specific dataset has similar performance on datasets from other domains. This is indicating that prompt tuning is leading to less domain overfitting. This is fundamentally different from data overfitting that when data overfitting is happening, the model has a better performance on the seen training data than unseen validation data that comes from the same dataset, whereas when domain overfitting is happening, the model has a better performance on the domain that is fine-tuned with, but this performance cannot easily be achieved when the model is zero-shot evaluated on this domain when it is fine-tuned on another domain, where both domains share exactly the same task and input & output formats.

4.2 GPT-2 & Table-to-Text Generation

Our experimentation results for prompt/prefix tuning on GPT2 model for table-to-text tasks are provided in Table 3. There are several discoveries that we have found under our experimentation settings:

First, prefix tuning is superior in all model sizes. The BLEU score of prefix tuning on the training/validation set is 30/15 points higher than prompt tuning for the base and large models. The best performance of the prefix tuned GPT2 model is comparable to the state-of-the-art method, which reaches around 67 BLEU scores. However, we observe a significant performance drop in training and validation set in prefix tuning but not in prompt tuning. This may be due to overfitting in-domain data, which also appears in the experiment on question answering. On the other hand, prompt tuning seems to underfit in-domain data. It learns the sentence structure well, but omits some keywords in the data. The examples in Table 4 show that prompt tuning misses a part of information in the source. This may explain why prompt tuning generalizes well in out-of-domain data.

Configurations		**In-Domain**				**Out-of-Domain**					
		Prompt		Prefix		Prompt			Prefix		
Size	# Tkns	B(S)	B(U)	B(S)	B(U)	B	T	M	B	T	M
Base	1	0	0	60.69	42.14	0	0.95	0.04	19.45	0.96	0.26
	5	30.01	24.16	62.51	**45.53**	28.32	0.66	0.2	**29.02**	0.75	**0.32**
	10	31.91	26.18	63.07	43.16	26.6	0.65	0.25	28.09	0.76	*0.32*
	20	37.17	33.8	**63.25**	44.9	27.91	0.62	0.27	16.45	1.63	0.31
	50	38.27	31.07	62.6	44.33	27	**0.61**	0.26	20.51	1.15	*0.32*
Large	1	0.69	0.88	64.02	45.91	0.44	0.97	0.04	22.7	1	0.32
	5	32.01	28.07	63.75	45.73	19.77	0.7	0.2	30.35	0.71	*0.34*
	10	35.86	32.25	63.97	**47.27**	20.67	0.8	0.21	30.23	0.71	*0.34*
	20	37.69	33.57	**64.44**	46.35	27.22	0.67	0.29	29.98	0.71	0.33
	50	40.17	36.85	64.23	46.43	24.61	0.79	0.3	**31.68**	**0.65**	**0.34**

Table 3: GPT-2 results on table-to-text generation task with prompt & prefix tuning. Here, webnlg dataset was used as the training set to train the model. The model is evaluated on the same (i.e. **S**een) training set, **U**nseen validation set with **B**LEU score from webnlg as in-domain performance. Then, the model is evaluated on DART dataset for out-of-domain performance in terms of **B**LEU, **T**ER, and **M**ETEOR score.

Second, there does not seem to have a number of tokens that consistently perform better. Although a token size of 20/50 gives the best performance on in-domain data, the performance on out-of-domain data could not be anticipated given a token length. Contrary to prompt tuning on T5, a token of length 1 performs worst in both prompt and prefix tuning on GPT2. In fact, our experiments show that the number of tokens needs to be larger than 3 for prompt tuning on table-to-text tasks.

Third, although prompt tuning performs better as the model size increases in-domain, it performs worse in a larger model out-of-domain. This unexpected degradation in performance does not appear in prefix tuning. More works need to be done in order to investigate the root cause.

Fourth, we obtain similar patterns that prompt tuning's performance on GPT2 diverges significantly giving different lengths of tokens, while prefix tuning's performance is relatively stable.

4.3 Common Patterns

The experiments on different lengths of tokens show that prompt tuning's performance diverges when having a different number of tokens. In contrast, prefix tuning's performance consistently keeps over different tokens for the same-sized model. This shows that the prompt length parameter in prompt tuning is critical.

Furthermore, prefix tuning seems to perform better in training and in-domain data. However, it unexpectedly yields worse results in out-of-domain data than prompt tuning when the model size grows. In contrast, prompt tuning performs comparable in out-of-domain to in-domain as the model size increases. This could be caused by overfitting in prefix tuning since it uses more parameters than prompt tuning. Still, both methods outperform fine-tuning in out-of-domain data.

5 Discussion

In this section, we discuss the advantage of prefix/prompt tuning and address some limitations in this study.

5.1 Advantages

Prefix and prompt tuning require much less time and resources to train while still obtaining comparable results to fine-tuning. Both methods only train on a small subset of parameters and freeze other parameters, significantly reducing training costs. Fewer parameters in prompt tuning may generalize even better in unseen and out-of-domain data. For example, its performance on the training and validation set is very close.

Prefix and prompt tuning are meaningful in real-life applications. For example, suppose we have many individual tasks but share the same model structure. Prefix and prompt tuning could maintain modularity and save time/space by only adding and deleting prefix/prompt tokens for each task. Beyond that, the inference is more efficient with prefix/prompt settings. Instead of having different models and calling forward multiple times, we can

<table>
<tr><td>Source</td><td>(Madrid : country : Spain <|endoftext|>, (country,))</td></tr>
<tr><td>Prefix tuning</td><td>Madrid is in Spain.</td></tr>
<tr><td>Prompt tuning</td><td>Madrid is the country of Spain.</td></tr>
<tr><td>Reference</td><td>Madrid is in the country of Spain.</td></tr>
<tr><td>Source</td><td>(Amsterdam_Airport_Schiphol : 5th_runway_SurfaceType : Asphalt <|endoftext|>, (5th_runway_SurfaceType,))</td></tr>
<tr><td>Prefix tuning</td><td>The 5th runway at Amsterdam Airport Schiphol is made of asphalt.</td></tr>
<tr><td>Prompt tuning</td><td>The 5th runway surface type is Asphalt.</td></tr>
<tr><td>Reference</td><td>The 5th runway at Amsterdam airport Schiphol has an asphalt surfacing.</td></tr>
<tr><td>Source</td><td>(Andrews_County_Airport : elevationAboveTheSeaLevel_(in_metres) : 973.0 <|endoftext|>, (elevationAboveTheSeaLevel_(in_metres),))</td></tr>
<tr><td>Prefix tuning</td><td>Andrews County Airport is 973 metres above sea level.</td></tr>
<tr><td>Prompt tuning</td><td>973.0 elevation above the sea level is 973.0 metres.</td></tr>
<tr><td>Reference</td><td>Andrews County Airport is 973 metres above sea level.</td></tr>
</table>

Table 4: Examples of generated table-to-text sentences. Red color represents the part of information that is missing in prompt tuning. Overall, prompt tuning could capture the sentence structure and the majority of the table information very well. Prefix tuning could capture all information and match the Reference.

do a single forward pass with batches.

5.2 Limitations

We have several limitations in the scope of this report. The direct comparison between prompt and prefix tuning is not very convincing. The hyperparameters in prompt tuning are not fine-tuned, but hyperparameters in prefix tuning experiments are tuned based on (Li and Liang, 2021). This directly causes prefix tuning to outperform prompt tuning in in-domain data. The implementation details of two methods are also slightly different. The implementation provided by the Prefix-tuning does not work on T5, so we modified the codebase, which may lead to minor discrepancies in implementations. The implementation of prompt tuning was not released when we started this project, so we built our pipeline, which is different from the official codebase. Our pretrained T5 model is also different from the one experimented in Lester et al. (2021)'s work.

Also, we do not perform ablation tests to examine the internal representation of prefix/prompt tokens. This is another exciting topic we want to explore in the future. For example, if we find some patterns in the space of prefix/prompt tokens, we could directly add a prefix/prompt to a pretrained model when a new task comes. This would allow us to obtain a model which has comparable performance to fine-tuned models, but with no extra costs.

6 Conclusion

We conclude that prompt tuning is more robust in domain-shift tasks. However, the length of prompt tokens is an important parameter and need to be tuned in different tasks. Because of time and resource limitations, our parameters are not fine tuned and the result is not perfect. We would like to further optimize the performance in in-domain data and see whether the score in out-of-domain also increases and achieves the same level.

On the other hand, prefix tuning does not generalize as good as prompt tuning in out-of-domain data, but its performance in in-domain data is close to the state-of-the-art fine tuning method. Furthermore, the prefix length has small affects in different tasks and model sizes. Hence, prefix tuning could reach fine tuning performance with much fewer parameters, less training time and less fine tuning process.

Acknowledgements

This report is an artifact of the Data Science Capstone Project at the University of California, San Diego. We thank our mentor Prof. Zhiting Hu for providing feedback and directions for the project. The computational resources are provided by the Data Science & Machine Learning Platform (DSMLP) at the University of California, San Diego. We also thank Yuan Gao, Yihan Wang from University of California, San Diego for providing some technical support at the very beginning of the project.

References

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Tom Bosc, Elena Cabrio, and Serena Villata. 2016. DART: a dataset of arguments and their relations on Twitter. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1258–1263, Portorož, Slovenia. European Language Resources Association (ELRA).

Yu Cao, Meng Fang, Baosheng Yu, and Joey Tianyi Zhou. 2020. Unsupervised domain adaptation on reading comprehension. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7480–7487.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855, Melbourne, Australia. Association for Computational Linguistics.

Emilie Colin, Claire Gardent, Yassine M'rabet, Shashi Narayan, and Laura Perez-Beltrachini. 2016. The WebNLG challenge: Generating text from DBPedia data. In *Proceedings of the 9th International Natural Language Generation conference*, pages 163–167, Edinburgh, UK. Association for Computational Linguistics.

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.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics.

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.

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.

Zhiting Hu and Li Erran Li. 2021. A causal lens for controllable text generation. In *Advances in Neural Information Processing Systems*.

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. 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, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in NLP—A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards Complex Language Understanding with Paraphrased Reading Comprehension. In *Meeting of the Association for Computational Linguistics (ACL)*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4596–4604. PMLR.

Matthew Snover, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231, Cambridge, Massachusetts, USA. Association for Machine Translation in the Americas.

Alon Talmor and Jonathan Berant. 2019. MultiQA: An empirical investigation of generalization and transfer in reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4911–4921, Florence, Italy. Association for Computational Linguistics.

Danqing Wang, Pengfei Liu, Ming Zhong, Jie Fu, Xipeng Qiu, and Xuanjing Huang. 2019a. Exploring domain shift in extractive text summarization. *CoRR*, abs/1908.11664.

Huazheng Wang, Zhe Gan, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, and Hongning Wang. 2019b. Adversarial domain adaptation for machine reading comprehension. 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 2510–2520, Hong Kong, China. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Ying Xu, Xu Zhong, Antonio Jose Jimeno Yepes, and Jey Han Lau. 2019. Forget me not: Reducing catastrophic forgetting for domain adaptation in reading comprehension. *CoRR*, abs/1911.00202. Withdrawn.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021. Crossfit: A few-shot learning challenge for cross-task generalization in nlp. *arXiv preprint arXiv:2104.08835*.

Rong Zhang, Revanth Gangi Reddy, Md Arafat Sultan, Vittorio Castelli, Anthony Ferritto, Radu Florian, Efsun Sarioglu Kayi, Salim Roukos, Avi Sil, and Todd Ward. 2020. Multi-stage pre-training for low-resource domain adaptation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5461–5468, Online. Association for Computational Linguistics.