CS 4803 Project Report: Adapting Transformers for Down-stream NLP Tasks

Yihe Liu, Lyndon Arthur Puzon, and Pavan Seshadri
Georgia Institute of Technology
{yliu3253,lpuzon6,pavan.seshadri}@gatech.edu

Abstract

Language models pretrained on large, varied corpora from several sources and domains serve as the backbone of much of modern Natural Language Processing research and applications. Standard practice for training Natural Language Classifiers involves using a pretrained model such as BERT, or GPT-3 and fine tuning the training to the target task and domain. Previous work has shown that both domain adaptive pretraining (DAPT) and task adaptive pretraining (TAPT), where a pretrained model is tuned to a specific domain source, and task's unlabeled data, respectively show SoTA performance. However, as these pretrained models may have potentially millions to billions of parameters, this becomes an extremely computationally heavy task. Adapter fine-tuning is a process in which trained layers are inserted within the frozen parameters of a pretrained model, dramatically reducing the learned parameters of the model. Previous work has shown promise in achieving SoTA performance using adapters. This study presents a survey of adapter-based pretraining methods for NLP classification tasks, over four domains (biomedical and computer science publications, news, and reviews), and across eight different classification tasks. Overall, we find that Adapter-based pretraining methods are able to reach similar, albeit marginally reduced performance for downstream NLP classification tasks.

1. Introduction

Today, to achieve state-of-the-art (SoTA) performance on a down-stream NLP task, a pretrained, Transformer-based language model is fine-tuned to the target task. Domain-adaptive pretraining (DAPT), where a pretrained model is tuned to a specific domain source, and task-adaptive pretraining (TAPT), where a pretrained model is tuned to a task's unlabeled data, are introduced in Guruangan *et al*. [3] as benchmark full fine-tuning methods for SoTA pretrained language models. However, full fine-tuning requires the generation of a new set of weights for each new task and/or domain, and since some SoTA language models are in the order of billions of parameters, this process can be both space and computationally expensive.

Houlsby *et al*. [4] describes a parameter-efficient alternative to full fine-tuning methods of SoTA language models that achieves SoTA performance: *adapter-based tuning*. Whereas full fine-tuning updates all the weights of the pretrained model, adapter-based tuning initializes a small set of weights at every level of the transformer and fine-tunes only those weights, while keeping the weights of the pretrained model fixed. This allows efficient training of task-specific adapters for the same model, which can be used as-is or combined with others. To facilitate the process of adapter-based tuning, Pfeiffer *et al*. [13] proposes an adapter training framework, *AdapterHub*. This framework is built on the HuggingFace `transformers` framework, allowing for intuitive training and exporting of adapters with only a few lines of code. Pfeiffer *et al*. [13] tested the AdapterHub implementation of two different adapter-based tuning methods on the GLUE benchmark, finding that adapters perform just as well as full fine-tuning.

1.1. Goals

We would like to compare the results of DAPT and TAPT on RoBERTa [8] for eight specific tasks as presented in Pfeiffer *et al*. [13] with adapter-based tuning on RoBERTa for the same eight tasks. The eight target task datasets and their appropriate specifications are in Table 1.

We will show that adapter-based tuning can perform just as well as DAPT or TAPT and cement adapter-based tuning as a more accessible, flexible, and less computationally complex strategy for fine-tuning SoTA pretrained language models.

2. Approach

2.1. Data

To directly compare to Guruangan *et al*. [3], each of the 8 target classification task datasets were datasets already previously tested for TAPT and DAPT. The datasets span the domains of Medical Research, Computer Science Research, News, and Reviews. Each dataset was split into

Domain	Task	Label Type	Train (Lab.)	Train (Unl.)	Val.	Test	Classes
BIOMED	CHEMPROT [7]	relation classification	4169	-	2427	3469	13
	RCT [1]	abstract sent. roles	180040	-	30212	30135	5
CS	ACL-ARC [5]	citation intent	1688	-	114	139	6
	SCIERC [9]	relation classification	3219	-	455	974	7
NEWS	HYPERPARTISAN [6]	partisanship	515	5000	65	65	2
	AGNEWS [15]	topic	115000	-	5000	7600	13
REVIEWS	AMAZON [11]	review helpfulness	115251	-	5000	25000	2
	IMDB [10]	review sentiment	20000	50000	5000	25000	2

Table 1: The target task datasets.

Dataset	Description
BookCorpus	Greater than 11,000 unpublished books
English Wikipedia	Wikipedia texts stripped of tables and headers
CC-News	Greater than 60 millions English news articles from 2016 to 2019
OpenWebText	WebText datasets used to train GPT-2
Stories	CommonCrawl data with story-like style

Table 2: ROBERTA: Number of data points and classes for each dataset

test/train/validation sets and preprocessed into embeddings by a pretrained ROBERTA tokenizer. For computational complexity reasons, each input embedding was truncated to a max length of 80 for every dataset other than RCT, AGNEWS and AMAZON, which used input embedding of max length 65 during training. Additional tests were performed on the CHEMPROT, RCT, AGNEWS, and IMDB datasets extending the max length to 512 to determine the effect of truncating the length.

2.2. Models

We used Adapter-Hub [13], based on the HuggingFace `transformers` and PyTorch frameworks to implement, train, and evaluate transformer adapters for each of the 8 target classification task datasets. The base pre-trained model used was ROBERTA [8], which an adapter and classification head was added to and trained for each task and dataset. Table 2 details the datasets and domains in which ROBERTA was trained on. Each trained adapter was added to the base pretrained ROBERTA with classification heads with output lengths of the number of classes per each dataset.

2.3. Training

Each model was trained for 100 epochs for the smaller datasets (train points $<$ 100K), and trained for 50 epochs for the larger datasets (train points $>$ 100K), due to the computational load. High Resource settings were used for these larger datasets. The learning rate was set to 1e-4, with a batch size of 12 for smaller datasets, and 120 for larger datasets. Cross entropy loss and Stochastic Gradient Descent were used as the loss function and optimizer, respectively. To mitigate differences between randomness during training, the random seed was manually set to be consistent across all runs.

2.4. Evaluation

To directly compare to Guruangan *et al*. [3], who compared baseline ROBERTA, DAPT, and TAPT using f1-macro and f1-micro scores, the same metrics were computed and compared for each trained adapter/dataset. Additional experiments were performed with the CHEMPROT, RCT, AGNEWS, and IMDB datasets using both input encoding length 80 and 512 to determine the effect on the maximum input length on performance.

2.5. Challenges

The main challenge we both faced and anticipated was the tradeoff between computational complexity and performance. Three of the eight datasets had over 100K training points, with varying sequence lengths, which frequently either attempted to take over twenty four hours for one full training session, and/or ran out of memory and disk space during training. Each member's access to compute resources was different, so a standardized training step/epoch and maximum sequence length was determined. For those

Domain	Task	RoBERTa	Finetune		adapter	
			DAPT	TAPT	DAPT	TAPT
BIOMED	CHEMPROT	81.9	**84.2**	82.6	82.0	82.7
	RCT	87.2	87.6	**87.7**	87.3	86.6
CS	ACL-ARC	63.0	**75.4**	67.4	65.1	70.6
	SCIERC	77.3	**80.8**	79.3	76.6	78.6
NEWS	HYPERPARTISAN	86.6	88.2	**90.4**	88.2	88.4
	AGNEWS	93.9	93.9	**94.5**	93.7	94.2
REVIEWS	Amazon	65.1	66.5	**68.5**	65.2	68.2
	IMDB	95.0	95.4	**95.5**	93.2	94.8

Table 3: Comparison of RoBERTa and DAPT and TAPT fine-tuning [3] to adapter-based DAPT and TAPT tuning.

		TAPT Adaptor	
	Sequence Length	80	512
BIOMED	CHEMPROT	82.7	**83.7**
	RCT	86.6	**87.2**
NEWS	AGNEWS	94.2	**94.6**
REVIEWS	IMDB	94.8	**95.0**

Table 4: Comparison of the performance of different sequence length inputs during training.

with lower access to compute resources, a sequence length of 80 and 65 was found to be the maximum lengths able to provide stable training for small and large data sets, respectively. To attempt to mitigate and quantify the performance loss from doing so, one member with access to powerful compute resources performed experiments using 80 and 512 maximum sequence lengths, respectively in order to better contextualize our results and quantify the difference.

3. Experiments and Results

Our adapted models are evaluated using macro-F1 metrics, except for CHEMPROT and RCT, for which we used micro-F1. The result are shown in Table 3. The scores in the RoBERTa and Finetune column are from Guruangan *et al*. [3], which followed the fine-tuning scheme for training. We report the performance of our adapted model in the adapter column, which includes DAPT and TAPT.

3.1. Comparison between DAPT and TAPT

Our results demonstrate that TAPT generally performs better than DAPT. This finding is also present in the results from the fine-tuning scheme. We speculate that TAPT's superior performance over DAPT derives from the closeness of its domains to the domain of the test dataset. However, even though model trained by TAPT experience less of a domain transfer, it also was trained on significantly less data than their DAPT counterparts. Thus, pretraining is crucial to the success of TAPT.

3.2. Comparison between fine-tuning and adapters

The best performing models in each domain and task are fine-tuned. This is expected as adapter is a cheaper and more efficient alternative to pretrain while producing comparable results [13]. Our adapter-based models do outperform the pretrained RoBERTa model on most tasks, demonstrating our method's capability to adapt.

We suspect that the difference in performance between fine-tuning and adapter is due to an adapter's limitation in the amount of parameter that it can change. During pretraining, all of the model's parameters can be changed, and therefore the model has a larger hypothesis space than that of an adapter-based model. Another reason why the adapter-based models performs marginally less than pretrained models is that we decreased the input sequence size to train on our hardware. Table 4 shows that the performance of adapter-based models can further increase when using longer sequence length as input.

4. Discussion

Our problem is deals with text-to-label processing. We used a transformer-based model to extract the sentence embedding from the input and a feed-forward layer as a classifier. The transformer model has a [CLS] token appended to the beginning of each input sequence. In the self-attention layers, the [CLS] takes information from different positions of the sequence. Therefore, the [CLS] token is usually used

to get a global representation or sentence embedding [14], while the output of each token in the input sequence can be seen as local representation or word embedding.

A transformer model's performance is dependent on a sufficient amount of training data. We reason this is because the transformer model has less handcrafted features than RNNs, which simulates the sequential manner in which human reads a paragraph of texts. The model has most of its learned parameters in the linear layers that projects the output of previous layers into Key, Value and Query for self-attention. The creation of the attention map itself is just a matrix multiplication step and does not require learned parameters.

The neural network expects sequence of word embeddings as inputs and its output format is based on its head. For classification tasks, the head can be just a linear layer and the loss function is Cross-Entropy. For sequence generation tasks, the head is a decoder. For Question Answering, the model learns two extra vectors of parameters that signifies the start and end of the answer. The attention between these two vectors and the model's outputs corresponding each input token entails the start and end position [2].

Throughout training, the model's performance in both train and test set increases, and the gap between them did not widen. We thus have reason to believe that the model did not overfit. The TAPT and DAPT approach can generalize quite well on domains and tasks with sufficient datasets.

5. Conclusions and Future Work

In this work we demonstrate the effectiveness of DAPT and TAPT methods with adapter-based tuning. Our experiments reveal that the usage of adapters can not only reduce the parameters of models and thus shorten training time, but also produce comparable results to that of fine-tuning. In addition, we discover the necessity to adapt transformer models to down-stream tasks for better performance. Future work to build on top of our results are data selection for TAPT [13] and fusion of adaptors to address multi-tasks [12].

References

[1] Franck Dernoncourt and Ji Young Lee. PubMed 200k RCT: a dataset for sequential sentence classification in medical abstracts. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 308–313, Taipei, Taiwan, Nov. 2017. Asian Federation of Natural Language Processing. 2

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. 4

[3] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of ACL*, 2020. 1, 2, 3

[4] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR, 09–15 Jun 2019. 1

[5] David Jurgens, Srijan Kumar, Raine Hoover, Dan McFarland, and Dan Jurafsky. Measuring the evolution of a scientific field through citation frames. *Transactions of the Association for Computational Linguistics*, 6:391–406, 2018. 2

[6] Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. SemEval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics. 2

[7] Jens Kringelum, Sonny Kim Kjaerulff, Søren Brunak, Ole Lund, Tudor I. Oprea, and Olivier Taboureau. Chemprot-3.0: a global chemical biology diseases mapping. *Database*, 2016, 2016. 2

[8] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019. 1, 2

[9] Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. 2

[10] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 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, June 2011. Association for Computational Linguistics. 2

[11] Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. *CoRR*, abs/1506.04757, 2015. 2

[12] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. *CoRR*, abs/2005.00247, 2020. 4

[13] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020): Systems Demonstrations*, pages 46–54, Online, 2020. Association for Computational Linguistics. 1, 2, 3, 4

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. 4

[15] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *CoRR*, abs/1509.01626, 2015. 2