Transformer Adapter Project: CS 7643 - Fall 2020
Team DRS

Dhruv Mehta
Georgia Institute of Technology
dmehta32@gatech.edu

Ruzbeh Irani
Georgia Institute of Technology
ruzbeh@gatech.edu

Sagar Jounkani
Georgia Institute of Technology
sjounkani3@gatech.edu

Abstract

The team will be working on the Transformer Adapter Capstone Project idea from Facebook. The project aims to replicate the results of Gurungan et al. (2020), which uses domain and task adaptation prior to fine-tuning to increase classification performance on a range of NLP tasks. We investigate the use of adapters and perform slightly better than the baseline results obtained by Gurungan et. al. for Task-Adaptive pretraining. We show that by using adapters, pre-trained models like RoBERTa base demonstrate an improvement in performance on low-resource classification tasks. Overall, adapters offer similar if not better performance at the cost of reduced training time and significantly less parameter-tuning.

1. Introduction

Transformer Adapters is a novel idea to use the existing transformer architectures and use transfer learning using different adapter modules to improve performance on various downstream NLP (Natural Language Processing) tasks. In the past, fine-tuning large pre-trained language models was viewed as an effective transfer mechanism for several downstream NLP tasks, however recent work has shown that using adapter modules on existing transformer architectures or pre-trained models are better suited and parameter efficient.

The goal of this project is to incorporate a language adapter another task adapter on the RoBERTa 2019 base model and compare task performance on two CS domain specific tasks to the results obtained by Gurungan et. al 2020 where the authors fine-tuned the RoBERTa base model for eight different tasks and four different domains. The work shown in this paper will compare and contrast performance for ACL-ARC task and SCI-ERC task in the CS domain.

2. Background and Motivation

The RoBERTa base model (Liu et. al 2019) is a pre-trained language model trained on over 160 GB of uncompressed text with various sources such as news articles, web content, literary text etc. The representations learned by this model are shown to have good performance on many downstream NLP tasks such as text-classification, question answering, NLI. However, recent work (Gurungan et. al 2020) shows transfer learning using a combination of fine-tuning and pre-training on domain/task specific corpus provides a significant boost in performance on the original language model.

The two-most common transfer learning techniques in NLP are feature-based transfer and fine-tuning for tasks. Features-based transfer involves pre-training real-valued embeddings vectors. These embeddings may be at the word (Mikolov et al., 2013), sentence (Cer et al., 2019), or paragraph level (Le Mikolov, 2014). The embeddings are then fed to custom downstream models. Fine-tuning involves copying the weights from a pre-trained network and tuning them on the downstream task. Recent work shows that fine-tuning often enjoys better performance than feature-based transfer (Howard Ruder, 2018). With standard fine-tuning, the pre-trained weights and a new top-layer are co-trained resulting in a whole new set of weights for each new task. For large models which can be 100s of MB in size, storing and sharing a complete set of weights for each task can be a challenge for multi-task applications.

Both feature-based transfer and fine-tuning require a new set of weights for each task. Fine-tuning is more parameter efficient if the lower layers of a network are shared between tasks. However, in the presence of many downstream tasks fine-tuning is parameter inefficient and an entirely new model is required for every task. The work done

by Housby et. al 2019 shows a new way of transfer learning using adapter modules. Adapter modules yield a compact and extensible model; they add only a few trainable parameters per task, and new tasks can be added without revisiting previous ones. The parameters of the original network remain fixed, yielding a high degree of parameter sharing. More recently, the work done by Pfeiffer et. al 2020 show that for solving classification tasks in NLP using an adapter-transformer framework to build and train domain and task specific adapters are much better suited and significantly outperform traditional strategies such as full fine-tuning of the model as well as multitask learning. Additionally, the authors also propose AdapterFusion, a framework for combining multiple adapters that are trained on different tasks and demonstrate that base language model BERT combined with this approach can achieve better performance than a single pre-trained and multi-task setup.

The project thus attempts to use and leverage the approaches described by Pfeiffer et. al 2020 and Housby et. al 2019 to use adapters with a RoBERTa base model as an alternate way of implementing task-adaptive pre-training (TAPT) in Gurungan et. al 2020. Task-adaptive pre-training (TAPT) refers to pretraining on the unlabeled training set for a given task. Gurungan et. al 2020 showed that task-adaptive pre-training (TAPT) on RoBERTa by way of fine-tuning followed by supervised training for the classification task gives better results than just supervised fine tuning on an off-the-shelf RoBERTa model. We will compare the results the authors obtained for two specific classification tasks namely the ACL-ARC task and the SCIERC task on the CS-domain for TAPT and augmented-TAPT and see if the approach using adapters has similar or better results.

The implications of our work would mean that given a resource-constrained environment on an NLP task, a framework using Adapter-Transformer architectures would be better suited and easier to adapt instead of using an approach involving fine-tuning. Additionally, pre-training can then be configured for just training the specific adapter module instead of the entire language model for either the task/domain.

3. Approach

3.1. Data

The task specific labelled datasets for supervised classification tasks namely, ACL-ARC task referred to as 'citation-intent' and SCIERC task referred to as 'sciie' was obtained from the original Gurungan et. al 2020 paper from their public repository https://github.com/allenai/dont-stop-pretraining/blob/master/environments/datasets.py

The data from the CS domain to perform augmented task-adaptive pre-training (TAPT) was sourced from the Kaggle open-source arXiv dataset https://www.kaggle.com/Cornell-University/arxiv which is a repository of over 1.7 million scholarly articles in multiple scientific fields. Both classification tasks ACL-ARC and SCIERC are based on scientific literature in the CS domain specifically relating to AI, Machine Learning (ML), Computational Linguistics, Information Theory, Information Retrieval and Computer Vision. Hence, for augmenting training data for TAPT we filtered for articles in the CS domain relating to these fields in the Kaggle open-source arXiv dataset. This dataset has been used for augmenting TAPT using adapters for both the tasks.The original dataset related to these papers consists of the entire papers and as such amounts to a large amount of text size which makes it infeasible to analyze and significantly increases training time for the adapter modules incorporated. Hence, in order to reduce the training time and allow feasibility for running code on an open-sourced platform like Google Colab, the original papers from this dataset were reduced by just extracting the abstracts of the same papers. Therefore, the dataset while reduced in terms of depth still maintains it original breadth of selection.

Finally, we use the publicly-available language model RoBERTa which can be downloaded and used from the following publicly available open-source repository https://github.com/allenai/dont-stop-pretraining.

3.2. Description

The team primarily used two publicly available repositories for this project, the first is the popular open-source transformer package supplied by HuggingFace at https://github.com/huggingface/transformers. This repository provides several APIs that allows one to download thousands of pretrained models to perform tasks on texts such as classification, information extraction, question answering, summarization, translation, text generation, etc in 100+ languages.

The second repository used is https://adapterhub.ml/ based on Pfeiffer et al 2020, which builds on the HuggingFace transformers framework allowing for easy training of adapters on a downstream NLP task. It fully supports the BERT and RoBERTa pre-trained language model with APIs for choosing the specific version of these language models.

Adapters serve the same purpose as fine-tuning a transformer based model but do it by stitching in layers to the main pre-trained transformer model, and updating the weights of these new layers, whilst freezing the weights of the pre-trained transformer model. An adapter module as described in Houlsby et al 2019 has a simple down-and-up-projection combined with a residual connection. We use the adapters as defined in the MAD-X framework in Pfeiffer et

al 2020 which classifies adapter modules into three types:

- **Language adapters** : a down-projection followed by a ReLU activation, an up-projection and a residual connection. These adapters have been shown to work with multi-lingual transformer models where these modules learn characteristics for one specific language and aid in adaptation to downstream tasks in this language. This is analogous, in our use-case, to learning characteristics of a specific domain and aid in downstream classification tasks in this domain.

- **Task adapters** : similar architecture as language models, these aim to capture knowledge that is task-specific but generalises across languages or domains. Natural language processing often involves sharing information across tasks. We often use something called Multi-Task Learning (MTL), but MTL suffers from two issues: 1) catastrophic forgetting: where 'information learned during earlier stages of training is "overwritten"' (Pfeiffer et al, 2020b). 2) catastrophic inference: where 'the performance of a set of tasks deteriorates when adding new tasks' (Pfeiffer et al, 2020b). With adapters, we train the task adapter for each task separately, meaning that we overcome both issues above.

- **Invertible adapters** : The invertible adapter has a similar function to the language adapter, but aims to capture token level language-specific transformations

This is where the use of the AdapterHub framework is essential for training the language adapter. Adapters serve the same purpose as fine-tuning but do it by stitching in layers to the main pre-trained model, and updating the weights ϕ of these new layers, whilst freezing the weights θ of the pre-trained model. Prior to this framework, stitching in adapters was difficult and required manual modification of the transformer architecture. With the use of this framework, we are able to dynamically "stitch-in" an adapter into the base language model. When we stitch-in adapters, we fix the representations of the rest of the transformer, which means these adapters are encapsulated and can be stacked or moved or combined with other adapters (Pfeiffer et. al 2020)[2].

3.3. Implementation

Below we highlight the steps we took in our approach and the order we undertook them:

Baseline: First step in our approach was to develop the baseline for both classification tasks ACL-ARC and SCI-ERC. Gururangan et al do it by supervised fine-tuning of an off-the-shelf RoBERTa base model for each classification task. We emulated this setup by training a task-adapter with a RoBERTa base model for each supervised classification task. Note that while training the task-adapter the parameters for the RoBERTa model are not updated through back propagation.

TAPT using language-adapters: Second step was to emulate TAPT as described in Gururngan et al by incorporating language-adapters in the transformers. Hence, instead of continuing pre-training the RoBERTa model via fine tuning as demonstrated by Gururngan et al, we continue pre-training via training only language-adapter parameters via masked-language-modelling (MLM) on task data and augmented task-domain data. The trained language-adapters act as replaceable modules within the transformer architecture which provide adaptation of an off-the-shelf RoBERTa model to the domain of the task data. We trained the following language-adapters, defined by their training data, configuration and adaptive pre-training type: *a*) ACL-ACR task data with Pfeiffer [1] configuration for TAPT. *b*) SCIERC task data with Pfeiffer configuration for TAPT. *c*) CS domain kaggle arXiv dataset [2] with Pfeiffer for augmented TAPT. *d*) CS domain kaggle arXiv dataset and modified Houlsby configuration for augmented TAPT.

Supervised training using task-adapters: Third step in our approach was to add task-adapters to the TAPT RoBERTa models for supervised training of the classification tasks. During this training phase we only updated the parameters of the task-adapter. We did an ablation study by running experiments with multiple configurations of the task adapter. Figure 1 shows how a specific adapter can be configured with different parameters. We varied three specific components of the task adapter namely **MH adapter, Output adapter and Invertible Adapter**. These parameters were chosen in order to understand the impact of position of adapter modules and advantage of using invertible adapters.

Exploring pre-trained adapters with fusion layers: We also experimented with adapter-fusion based on the approach described in Pfeiffer et. al 2020[2]. We trained a fusion layer for the supervised classification tasks using the adapterhub framework for the language-adapters and the baseline task-adapter.

4. Experiments

The goal of the approach described was to compare the results of using an adapter based framework against the results of fine-tuning as presented in the Gurungan paper (2020). In the below sections, we outline the experiments and results and compare them to the paper results which serve as a baseline.

[1] Pfeiffer[2] and Houlsby[1] are two adapter configurations provided with the HuggingFace Transformers library. We modified the Houlsby configuration by appending an invertible adapter to it for masked language modelling. Both the configs are based on their respective papers.

[2] Refer to section 3.1 for information on dataset

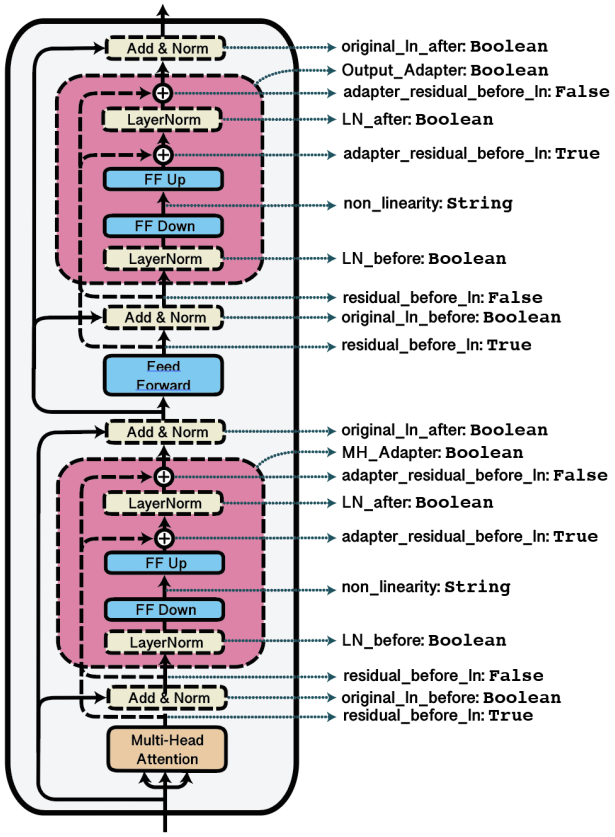

Figure 1. Adapter Configurations for Ablation Study Inforgraphic

4.1. Baseline

For a baseline model we trained a task adapter and classification with the RoBERTa model. Parameters for the RoBERTa model were frozen. The results shown below demonstrate that our baseline with the task adapter improves on the results from Gururangan et al[3] for both classification tasks.

RoBERTa Baseline		
Task Name	Team DRS	Gurungan Results
ACL-ARC	$65.3_{2.3}$	$63.0_{5.8}$
SCIERC	$82.2_{0.9}$	$77.3_{1.9}$

4.2. Task-Adaptive Pre-Training

For task-adaptive pre-training we trained a language adapter with Pfeiffer[2] configuration for each of the 2 two classification tasks using the labelled training data provided by Gururangan et al[3]. We used a batch size of 32 with gradient accumulation for 2 steps and trained the models for 100 epochs with learning rate of 10e-4. The masked word probability for the MLM training was 0.15. MLM masks words with a probability and the task for the model is to learn the distribution over the vocabulary for the masked words . Loss used for this task is categorical cross-entropy.

For evaluation of classification performance, a task-adapter with a classification head was trained with the TAPT language-adapter and the RoBERTa parameters frozen, for both supervised classification tasks (ACL-ACR and SCIERC). The task-adapter configuration we used was Pfeiffer[2], batch-size of 4 and 20 epochs with a learning rate of 10e-4. We experimented with various batch sizes and selected the one with the best validation performance.

The results are summarized in the table below

TAPT Results		
Task Name	Team DRS	Gurungan Results
ACL-ARC	$65.1_{3.3}$	$67.4_{1.8}$
SCIERC	$81.9_{0.7}$	$79.3_{1.5}$

4.3. Augmented TAPT (Ablation Study)

As described in our approach, we used the Kaggle arXiv dataset for augmenting TAPT. We filtered around 160K scientific abstracts from over 1.7 million papers. These abstracts were in the CS domain particularly in the field of AI, ML, Linguistics and Information Theory. This filtering was to emulate the task-domain data selection (CS domain for us) done by Gururangan et al[3] using Vampire model. We trained a language adapter with an off-the-shelf RoBERTa base model on this data for augmented TAPT. This language-adapter acted as a common module for adaptive pre-training for both classification tasks.

We used a batch size of 64 and trained the language-adapter for 30 epochs with a learning rate of 10e-4. The configurations we used were Pfeiffer[2] and modified Houlsby1] (we appended an invertible adapter to the Houlsby[1] configuration for enabling masked language modelling)

The evaluation of classification performance using the augmented language-adapter is similar to the setup for TAPT. The task-adapter with a classification head was trained with the augmented TAPT language-adapter and the RoBERTa parameters frozen.

We experimented with multiple configurations for the task adapter by varying the position and use of invertible adapter. Based on position adapters are defined as follows: *a*) `Multi-Head (MH) Adapter` is an adapter after the multi-head attention layer and before the feed-forward layer. *b*) `Output Adapter` is an adapter after the feed forward layer and followed by an add-and-norm layer (residual connection and layer norm) which produces the transformer output.

The best validation macro F1 for both tasks was with an MH adapter and an invertible adapter activated. The table below shows the best result selected for comparison.

Augmented TAPT Results		
Task Name	Team DRS	Gurungan Results
ACL-ARC	$65.3_{2.0}$	$75.4_{2.5}$
SCIERC	$82.3_{0.9}$	N/A

Results for all task-adapter configurations are available in the appendix. While the above do not show much improvement over our baseline in comparison to Gururangan et al[3] had a 2-4% improvement across both tasks using fine-tuning. The reasons we think adapters haven't performed that well for TAPT in our low-resource tasks (ACL-ARC and SCIERC) is lack of enough data for training an effective language adapter from scratch. Compared to fine-tuning RoBERTa, which already performed decent off-the-shelf on these tasks, we trained an adapter module for task language representation from randomly initialised weights. Our hypothesis is that a language adapter requires more data to be as effective as fine tuning. We tested this hypothesis with augmented TAPT.

For augmented TAPT, we have slight improvement over our baseline but not comparable to the improvement for Gururangan et al[3]. The improvement can be attributed to the more training data for the language adapter but only more data is probably not enough and more targeted augmentation of data for TAPT is what is required for a meaningful increase in performance. We selected augmentation data using arXiv paper categories like AI, ML and Information Theory and then random sampled from this filtered dataset. In comparison the experiments from Gururangan et al[3] use human curated TAPT and automated data selection for TAPT using the VAMPIRE model.

5. Conclusion

We investigated the use of adapters under various configurations, as an alternative to fine-tuning for adapting pre-trained LMs to a particular task domain. Our results show that task-adapters can help in improving results on large pre-trained models like RoBERTa. While our results do not emulate the benefits for TAPT shown in Gururangan et al[3] in the low-resource classification tasks, it may be valuable to investigate as a future research direction: *a*) adapter configurations which can work in low-resource setting. *b*) better data selection for TAPT in low-resource setting *c*) the viability of adapters compared to fine-tuning in high-resource setting

Adapters offer a very exciting avenue of research for portable, modular and effective ways of adapting large pre-trained models to distant domains.

6. Work Division

A brief overview of the contribution of each team member to the project is provided below.

Sagar Jounkani :Research, Data creation, Implementation, Analysis. *Details* : Augmented TAPT, Lang. adapter training, Task adapter training, Experimentation, Paper writing

Ruzbeh Irani : Research, Implementation, Analysis. *Details* : Lang. Adapter Codebase, Task Adapter Training and execution, Experimentation and paper writing

Dhruv Mehta : Research, Implementation, Analysis, Ablation Study *Details* : Task Adapter Training, Ablation Study Experiments, Paper Writing, Adapter Fusion

7. References

1. Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, Sylvain Gelly, *"Parameter-Efficient Transfer Learning for NLP"*, In ICML 2019.

2. Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath,Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, Iryna Gurevych, *"AdapterHub: A Framework for Adapting Transformers"*, arXiv preprint.

3. Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, Noah A. Smith, *"Don't Stop Pretraining: Adapt Language Models to Domains and Tasks"*, In ACL 2020.

4. Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, Iryna Gurevych, *"AdapterFusion: NonDestructive Task Composition for Transfer Learning"*, arXiv preprint.

5. 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:1907.11692

8. Appendix

The entire code repository can be found at this link: https://github.gatech.edu/rirani6/cs7643_DRS

8.1. Ablation Study Results

Roberta Base - No Language Adapter

Base RoBERTa - No Lang Adapter		
Task Name	Adapter Config - MH adapter, Output Adapter Invertible adapter	F1 Score
ACL-ARC	False, True, False	$63.3_{0.9}$
ACL-ARC	True, False, True	$66.3_{0.3}$
ACL-ARC	True, False, False	$64.7_{0.4}$
ACL-ARC	False, True, True	$67.8_{2.5}$
SCIERC	False, True, False	$81.8_{0.6}$
SCIERC	True, False, True	$81.9_{0.1}$
SCIERC	True, False, False	$81.1_{0.1}$
SCIERC	False, True, True	$81.9_{0.6}$

Roberta Base + Augmented TAPT (Pfeiffer Config)

Base RoBERTa + Augmented TAPT (Pfeiffer Config)		
Task Name	Adapter Config - MH adapter, Output Adapter Invertible adapter	F1 Score
ACL-ARC	False, True, False	$63.5_{1.1}$
ACL-ARC	True, False, True	$64.8_{2.6}$
ACL-ARC	True, False, False	$62.8_{2.2}$
ACL-ARC	False, True, True	$63.5_{1.1}$
SCIERC	False, True, False	$81.9_{0.5}$
SCIERC	True, False, True	$81.8_{0.5}$
SCIERC	True, False, False	$82.4_{0.5}$
SCIERC	False, True, True	$82.3_{0.9}$

Roberta Base + Augmented TAPT (Houlsby Config)

Base RoBERTa + Augmented TAPT (Pfeiffer Config)		
Task Name	Adapter Config - MH adapter, Output Adapter Invertible adapter	F1 Score
ACL-ARC	False, True, False	$65.3_{2.0}$
ACL-ARC	True, False, True	$63.7_{1.3}$
ACL-ARC	True, False, False	$63.1_{2.7}$
ACL-ARC	False, True, True	$65.3_{2.0}$
SCIERC	False, True, False	$78.9_{1.5}$
SCIERC	True, False, True	$81.8_{0.4}$
SCIERC	True, False, False	$80.5_{0.5}$
SCIERC	False, True, True	$81.5_{0.4}$

8.2. Language-Adapter MLM training

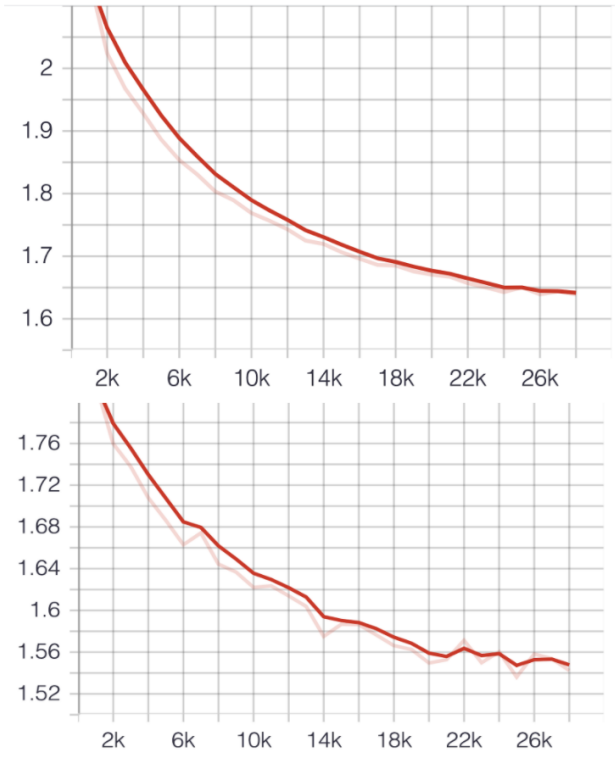

Figure 2. Pfeiffer Train & Eval loss (Aug. TAPT)

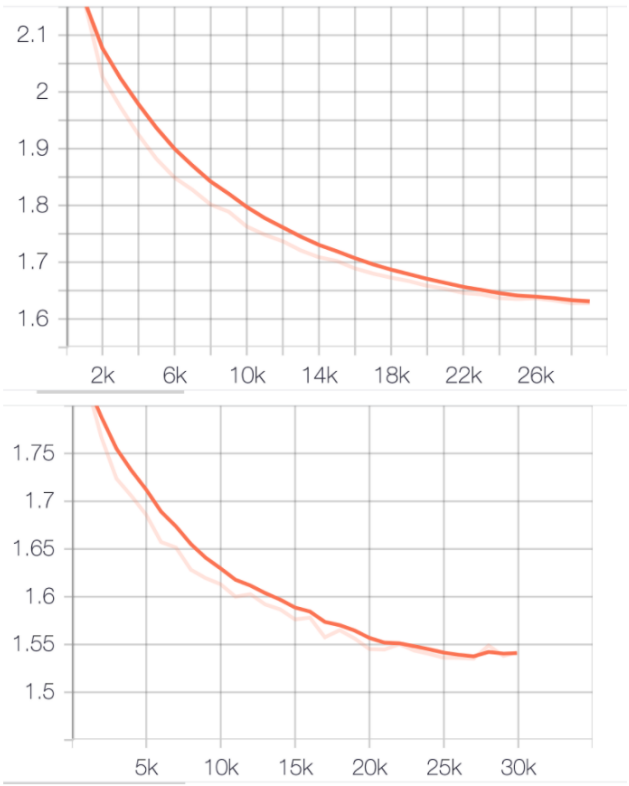

Figure 3. Houlsby Train & Eval loss (Aug. TAPT)