Improving Generalization of Adapter-Based Cross-lingual Transfer with Scheduled Unfreezing

Anonymous ACL submission

Abstract

Standard fine-tuning of language models typically performs well on *in-distribution data*, but suffers with generalization to *distribution shifts*. In this work, we aim to improve generalization of adapter-based cross-lingual task transfer where such cross-language distribution shifts are imminent. We investigate scheduled unfreezing algorithms—originally proposed to mitigate catastrophic forgetting in transfer learning—for fine-tuning task adapters in cross-lingual transfer. Our experiments show that scheduled unfreezing methods close the gap to full fine-tuning and achieve state-of-the-art transfer performance, suggesting that these methods can go beyond just mitigating catastrophic forgetting. Next, aiming to understand these empirical findings, we investigate the learning dynamics of scheduled unfreezing using Fisher Information. Our experiments reveal that scheduled unfreezing induces different learning dynamics compared to standard fine-tuning, and provide evidence that the dynamics of Fisher Information during training correlate with cross-lingual generalization performance. We additionally propose a general scheduled unfreezing algorithm that achieves an average of 2 points improvement over four datasets compared to standard fine-tuning and provides strong empirical evidence for a theory-based justification of the heuristic unfreezing schedule: i.e., the heuristic schedule is implicitly maximizing Fisher Information. Our code will be publicly available.

1 Introduction

In the standard cross-lingual task transfer setup, a typical and often valid assumption is that only English data is available for fine-tuning and validation of a pretrained multilingual model, due to resource constraints in many languages (Hu et al., 2020). The trained model needs to generalize well to text inputs provided in other languages: this requirement can be seen as an extreme case of *distribution shift* generalization (Ramponi and Plank, 2020).

Parameter-efficient fine-tuning methods such as adapters (Houlsby et al., 2019; Stickland and Murray, 2019; Bapna and Firat, 2019) with separate language and task components are often used to achieve effective cross-lingual transfer, especially to low-resource languages (Pfeiffer et al., 2020, 2021; Ansell et al., 2021; Parović et al., 2022). These adapters insert a small number of trainable parameters into a frozen pretrained multilingual language model (e.g., mBERT, Devlin et al. 2019; XLM-R, Conneau et al. 2020) to achieve positive transfer while avoiding catastrophic forgetting (CF, McCloskey and Cohen 1989) of previously learnt knowledge after adapting to new tasks.[1] In other words, adapters enable *catastrophic forgetting free* learning, referred to as *CF-free* in this paper. However, while efficient, the adapter methods often incur a cross-lingual performance gap compared to full model fine-tuning.

Gradual unfreezing (GU) is a technique which unfreezes layers of deep neural models from top to bottom during training (Howard and Ruder, 2018). GU was previously proposed for general transfer learning for in-distribution data in monolingual contexts in NLP, and has predominantly applied to full fine-tuning. More recently, 'Linear-Probing-then-Fine-Tuning' (LPFT, Kumar et al. 2022) was proposed for transfer learning of both in-distribution and distribution-shifted evaluation data using full fine-tuning in computer vision. LPFT first trains the classification layer only, and then the full model. The main notion connecting these methods is training different layers of a neural network by unfreezing layers at different times (i.e., with a *schedule*). Designed to mitigate CF, these '*scheduled unfreezing*' methods have shown promising transfer learning results. However, it is unclear whether scheduled unfreezing can can do more than just mitigate CF, benefiting CF-free methods and

[1] Adapters bypass this issue since the pretrained model is kept frozen with its original weights unchanged.

cross-lingual transfer as a different type of distribution shift than previously studied.

In this work, we begin by asking the following question: Do scheduled unfreezing methods improve cross-lingual transfer and close the gap to full fine-tuning in the CF-free setting? We use scheduled unfreezing to train task adapters following the standard adapter-based cross-lingual transfer setup of Pfeiffer et al. (2020). We find that scheduled unfreezing acts as a generalization booster and closes the gap to full fine-tuning, confirming our hypothesis that, since cross-lingual transfer can be seen as a form of distribution shift, methods such as GU are effective, even in the CF-free setting. Our results suggest that there indeed is more to the original scheduled unfreezing training than just mitigating catastrophic forgetting (Howard and Ruder, 2018).

We further analyze the learning dynamics of scheduled unfreezing, with a particular focus on GU, using the trace of the Fisher Information Matrix (Kleinman et al. 2022, termed $\mathrm{tr}(F)$ henceforth). Our experiments reveal **1)** that scheduled unfreezing changes the dynamics of $\mathrm{tr}(F)$ during training, **2)** that $\mathrm{tr}(F)$ is a potential proxy for studying cross-lingual generalization.

Based on our analysis, we then propose an automatic scheduled unfreezing algorithm based on maximizing the $\mathrm{tr}(F)$ (termed **F**isher **Un**freezing or FUN), to generalize from previous heuristic-based methods. FUN achieves comparable results to heuristic-based methods and provides concrete empirical evidence that GU implicitly maximizes $\mathrm{tr}(F)$ during training in our experimental setting.

Contributions. In sum, our contributions are as follows. **1)** To the best of our knowledge, we are the first to demonstrate that scheduled unfreezing in task-adapter training for cross-lingual transfer closes the performance gap to full model fine-tuning. **2)** We present a generalized scheduled unfreezing framework that encompasses several existing methods, allowing easy extensions to new algorithms. **3)** As an analytical contribution, we demonstrate that Fisher Information is an effective tool for studying generalization in cross-lingual transfer and find that the dynamics of Fisher Information correlate with cross-lingual transfer results. **4)** We propose a $\mathrm{tr}(F)$-based scheduled unfreezing method (FUN), which achieves comparable performance to heuristic methods. FUN allows us to show that GU achieves good generalization in the CF-free setting by implicitly maximizing $\mathrm{tr}(F)$.

2 Related Work

Scheduled Unfreezing. Howard and Ruder (2018) propose Gradual Unfreezing (GU) to mitigate catastrophic forgetting when transferring a pretrained model for monolingual downstream tasks in non-Transformer architectures. Raffel et al. (2022) study GU for transferring full fine-tuning Transformers to in-distribution tasks, and empirically conclude that GU may not be effective. Recently, Kumar et al. (2022) (LPFT, Linear Probing then Fine-Tuning) show promising results for distribution-shifted transfer in the full fine-tuning setting for computer vision tasks. Yang et al. (2022) extend LPFT for full fine-tuning to NLP. Lee et al. (2022) show that fine-tuning different parts of a network helps with generalization under different types of distribution shifts in computer vision. Different from our work, the prior work has not studied scheduled unfreezing methods for cross-lingual transfer in a CF-free setting.

Fisher Information and Generalization. Fisher Information (Fisher, 1925) is an established concept in optimization theory and practice (Amari, 1998), e.g., to measure parameter importance (Kirkpatrick et al., 2017) or for pruning (Singh and Alistarh, 2020). Achille et al. (2019) study learning dynamics[2] in neural network training with Fisher Information. Golatkar et al. (2019) show Fisher Information correlates with generalization in computer vision and non-Transformer architectures. Jastrzebski et al. (2021) propose to regularize Fisher Information during the training of a neural network for better generalization in computer vision. Xu et al. (2021); Sung et al. (2021) create sparse masks using Fisher Information for better parameter-efficient tuning. We study Fisher Information in adapters and for cross-lingual transfer, which has the potential to guide the understanding of new methods in this area.

3 Methodology

3.1 Adapter-Based Cross-lingual Transfer

Adapters are lightweight components that are inserted into a multilingual pretrained Transformer model (termed the *base model*) to specialize the model for a particular purpose, such as adapting the large base model to a specific language (Pfeiffer

[2]Different from training dynamics (Swayamdipta et al., 2020) which focus on the data, learning dynamics focus on the model and the optimization process.

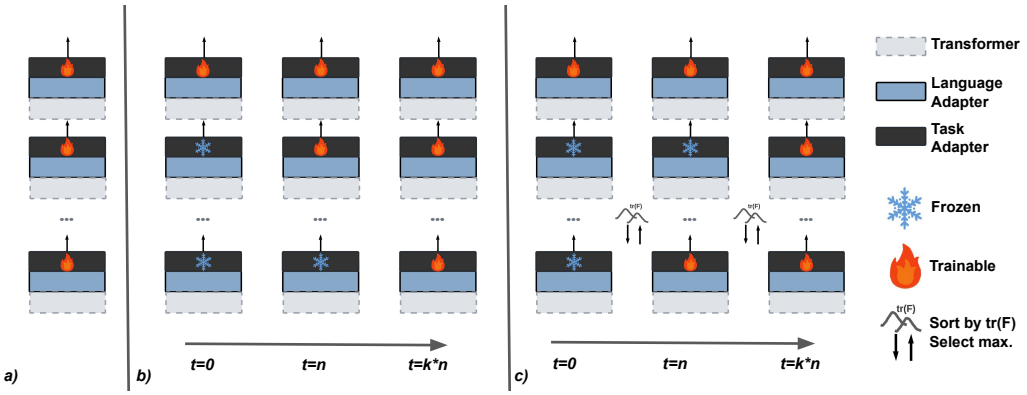


Figure 1: **a)** Standard, **b)** Gradual unfreezing versus **c)** $\mathrm{tr}(F)$-based scheduled unfreezing for training task adapters in adapter-based cross-lingual transfer. The classifier is not shown and is always trainable. All other components excluding task adapters, such as the original parameters of the base model and language adapters, are always frozen.

et al., 2020) or a task (Houlsby et al., 2019).

The adapter-based fine-tuning process typically spans two stages for cross-lingual *task* transfer. First, different language adapters are trained (often separately, per language) with the base model frozen, using the masked language modeling (MLM) objective in a target language. Second, task adapters and a task-specific output head are randomly initialized and inserted into the base model along with now-trained source language (i.e., typically *English*) adapters. In this stage, only the task adapters and a task-specific output head are trainable (by a task-specific loss), while all other parameters are kept fixed. At inference time, the source language adapters are replaced by the language adapter of the target language, while the task adapter is retained, to achieve zero-shot cross-lingual transfer. In this work, we base our experiments on the state-of-the-art cross-lingual adapter framework: MAD-X (Pfeiffer et al., 2020). Each adapter consists of a down-projection followed by a ReLU activation and an up-projection, inserted after the feed-forward layer in every Transformer layer. See Figure 6 in Appendix A for details.

3.2 Gradual Unfreezing and General Scheduled Unfreezing

First proposed by Howard and Ruder (2018), gradual unfreezing (GU) tunes a subset of parameters of a pretrained model starting from the top layer (see Figure 1b). Given a model with L layers, assuming that the index $L-1$ refers to the top layer, and interval k, GU unfreezes each layer starting from $L-1$ to 0 in order, every k steps. Once a layer is unfrozen, it remains unfrozen. Hence, the number of trainable parameters increases every k steps under the GU regime.

Algorithm 1 Generalized Scheduled Unfreezing

Require: An L-layer model with layer $j \in \{0, \ldots, L-1\}$ parameterized by θ_j. An additional task-specific classification head C. Total training budget N. Training interval k. Typically $kL \ll N$ for convergence.

```
Initialize C, θ_j for all j
S ← {C}
for i = 1 ... N do
    Sample a data batch b ~ D
    if i mod k == 0 and i ≤ kL then
        J = SELECT(*)    ▷ Set of layer (task adapter) indices to unfreeze.
        S ← S ∪ {θ_j : j ∈ J}
    end if
    FORWARD(*)
    for t = 1 ... |S| do
        GRADIENT_UPDATE(θ_t)
    end for
end for
```

Let SELECT$(*)$ be a layer-selection function. Let FORWARD$(*)$ be the standard forward pass through all layers, and GRADIENT_UPDATE$(*)$ calculates gradients and performs updates for parameters. We define a *generalized scheduled unfreezing* algorithm that encompasses GU, LPFT, and even the recent Surgical fine-tuning method (Lee et al., 2022), as well as the other variants we propose later in this work, in Algorithm 1.[3]

3.3 Fisher Information

We use the *Fisher Information Matrix* (F) to investigate changes in learning dynamics. Recent studies have shown that Fisher Information Matrix correlates well with the generalization capabilities

[3]For LPFT, SELECT$(*)$ returns $\emptyset$ for the first k steps and $\mathcal{J} = \{\theta_j\}$ for all layer indices j after the first k steps. In this work, we restrict ourselves to uniform time-intervals and leave the exploration of non-uniform intervals to future work.

of neural models (Golatkar et al., 2019; Jastrzebski et al., 2021). Conveniently, as a 2nd-order metric based on gradients, F also provides insights into the optimization process.

In particular, we take the trace of F (i.e., $\mathrm{tr}(F)$), since the full F is computationally expensive to obtain, and previous work has shown the $\mathrm{tr}(F)$ correlates well with F and shows similar general trends as the full F (Achille et al., 2019). Let x be data input and consider a network parameterized by weights w that encodes the approximate posterior distribution $p_w(y|x)$. The $\mathrm{tr}(F)$ is computed using the empirical data distribution $\hat{Q}(x)$ as follows:

$$\mathrm{tr}(F) = \mathbb{E}_{x\sim\hat{Q}(x)}\,\mathbb{E}_{y\sim p_w(y|x)}\,||\nabla_w \log p_w(y|x)||^2 \quad (1)$$

Note that Eqn. (1) is not the "empirical Fisher" (Kunstner et al. 2019, empirical Fisher uses the true data label y). Hence, we do not need the labels of input data as the y for calculating the true F. They are sampled from the label distribution of the task (i.e., $y \sim p_w(y|x)$, pseudocode in Appendix D).

One interpretation of F is that given w and a perturbed version of w' (after applying one step of gradient descent, for example), the KL divergence between $p_w(y|x)$ and $p_{w'}(y|x)$ is given by $\delta w \cdot F\delta w + O(\delta w^3)$ (up to 2nd-order approximation, where δw is the small perturbation in weights) (Martens, 2020).

F can be considered as a measure of how much a change in weights can affect the network output (i.e., how much information resides in the weights). Intuitively, this means a set of weights with near-zero entries in F likely means they do not significantly affect the network output (and thus task performance). Moreover, F is also a 2nd-order approximation of the Hessian of the loss function (Shun-ichi and Hiroshi, 2000; Martens, 2020) and provides information on the curvature of the loss landscape near the current weights, that is, how fast the gradients change during the optimization.

4 Experiments

4.1 Models

Base Models. The main experiments are conducted with two established pretrained multilingual models: mBERT (base-cased, Devlin et al. 2019) and XLM-R (base, Conneau et al. 2020).

Adapters. (^{Ada}) We follow the adapter configurations from MAD-X (Pfeiffer et al., 2020) for cross-lingual transfer, see §3.1. We use pretrained language adapters available in the AdapterHub.[4]

Scheduled Unfreezing Methods. We apply and analyze two scheduled unfreezing methods from research in other areas of NLP and computer vision to the task of adapter-based cross-lingual transfer:

LPFT (Kumar et al., 2022) (**+LPFT**) Originally proposed for full fine-tuning computer vision models, where the training process first trains the classification head (linear probing, LP) with the base model frozen, then unfreezes the entire model for fine-tuning (FT). In our setup, we first fine-tune the classifier, then followed by a step of unfreezing the adapters all at once for further fine-tuning.

Gradual Unfreezing (Howard and Ruder, 2018) (**+GU**) performs top-down unfreezing during training or fine-tuning. We fine-tune with the classifier and the top-most adapter unfrozen, and for every k steps we unfreeze the next adapter and continue.

4.2 Datasets and Hyperparameters

We conduct experiments on a diverse set of cross-lingual transfer tasks and target languages. We use MLQA (Lewis et al., 2020) and XQuAD (Artetxe et al., 2020) as evaluation datasets for question answering (SQuAD, Rajpurkar et al. 2016 for training). We use XNLI (Conneau et al., 2018) as the evaluation data for natural language inference (training on MNLI, Williams et al. 2018), and we use XCOPA (Ponti et al., 2020) for evaluating causal commonsense reasoning (training on COPA, Roemmele et al. 2011). The data statistics and language codes are summarized in Appendix C. We experiment in the zero-shot setting with English-only task data for training and validation.

Hyperparameters. We perform a hyperparameter search with the learning rates of [1e-4, 2e-4, 5e-4, 8e-4] for adapter fine-tuning on all datasets except COPA. For COPA, we found a much smaller learning rate (1e-5) is better for scheduled unfreezing methods (not standard training). For scheduled unfreezing experiments, we search for the hyperparameter k in the following range [25, 50, 100, 800, 1000] except for the experiments with smaller training data; see Appendix B for detailed hyperparameters. The reported results are averaged across 4 runs on A100 or V100 GPUs.

[4]There are missing language adapters from the AdapterHub for mBERT; we have thus trained our own language adapters following the AdapterHub recommendations for hyperparameter values. Please see Appendix B for details.

5 Results and Analysis

5.1 Scheduled Unfreezing Helps with Cross-Lingual Generalization

Figure 2 shows the relative performance of (a) task adapters fine-tuned in the standard way (^{Ada}), (b) GU- (^{Ada}+GU) and LPFT-tuned adapters (^{Ada}+LPFT) compared to full fine-tuning with mBERT and XLM-R on all datasets.[5] We report the results averaged across all target languages in the respective datasets. Our experiments show that both LPFT and GU are effective in closing the gap to full fine-tuning across all tasks and models. Moreover, GU-trained task adapters perform better, even exceeding the performance of full fine-tuning in some cases.[6]

Our results suggest that scheduled unfreezing can do more than just mitigate catastrophic forgetting. Even in a CF-free setting like ours, they achieve better generalization for cross-lingual transfer. We focus on GU as the scheduled unfreezing method for further analyses, since it produced better empirical results than LPFT.

Influence of Task Training Data Size. The training data for both XQuAD and XNLI are well over 50k instances; this amount of annotated task data might be unrealistic for some other tasks in practice, even in English. In order to simulate setups with fewer annotated data for training and analyze the impact of the effect of scheduled unfreezing in those setups, we sample 1k, 5k, and 10k training examples from SQuAD and MNLI.

We evaluate GU against the standard adapter training baseline (Table 1). With a smaller amount of training data, we still observe the advantages of GU over standard task adapter fine-tuning.

5.2 Scheduled Unfreezing Beyond Mitigating Catastrophic Forgetting

To understand why scheduled unfreezing helps even in the CF-free setting, we examine the learning dynamics during the training of task adapters.

Due to unfreezing of task adapters at different times, the model has access to a different number of trainable parameters, which affects the optimization and information encoding for adapters differently under scheduled unfreezing than in standard fine-tuning. Hence, we draw our attention to higher-order metrics, captured in $\mathrm{tr}(F)$.

XQuAD (F1)	1K	5K	10K
mBERT^{Ada}	45.27±0.59	52.58±0.81	55.89±1.08
mBERT^{Ada}+GU	**45.93**±0.50	**53.10**±0.35	**56.47**±0.74
XLM-R^{Ada}	44.11±1.43	57.20±0.36	61.75±0.68
XLM-R^{Ada}+GU	**48.42**±1.20	**59.88**±1.51	**65.28**±0.76
XNLI (Accuracy)	1K	5K	10K
mBERT^{Ada}	43.86±1.43	49.68±0.73	52.34±0.40
mBERT^{Ada}+GU	**44.69**±0.61	**51.67**±0.43	**53.95**±1.47
XLM-R^{Ada}	52.75±2.03	**64.22**±1.04	65.80±0.61
XLM-R^{Ada}+GU	**52.86**±1.38	64.15±0.35	**65.91**±0.56

Table 1: Cross-lingual transfer performance with subsampled English task data for task fine-tuning.

GU Changes Learning Dynamics. We plot $\mathrm{tr}(F)$ (moving average, normalized by the number of trainable adapters at the given step) during optimization. Figure 3 shows $\mathrm{tr}(F)$ during training on three datasets.[7] The plots indicate that GU significantly changes the learning dynamics of adapters compared to their standard fine-tuning. With GU, due to the model having fewer parameters to encode the same amount of data initially, the $\mathrm{tr}(F)$ curve is higher than in standard fine-tuning. The training process induces a $\mathrm{tr}(F)$ curve that has a distinctive "hill" shape (i.e., a "learning period", with fast changes of gradients, and hence weights).

Effects of the Unfreezing Schedule on $\mathrm{tr}(F)$ and Generalization. While we have empirically validated that scheduled unfreezing changes $\mathrm{tr}(F)$ during learning, it is unclear which factors are the main drivers that impact $\mathrm{tr}(F)$, and potentially improve generalization. Previous work has studied factors such as learning rate, weight decay, optimizer and loss functions (Golatkar et al., 2019; Jastrzebski et al., 2021). However, based on this work, another novel relevant factor is the *unfreezing schedule* (i.e., the number of parameters to update at each optimization step). Here, we aim to further understand how unfreezing schedules change the learning dynamics and generalization.

We found that the sensitivity of the learning dynamics to schedules depends on the dataset and the base model. Hence, we focus on XLM-R for MLQA/XQuAD and mBERT for XNLI. These two settings are the most sensitive to schedules and best illustrate the effects, but they are different in terms of models and tasks to examine general patterns.

[5] Baseline full parameter fine-tuning results used for Figure 2 are in Table 7 of the Appendix.

[6] Although we arrived at a different conclusion from Raffel et al. 2022, we emphasize that Raffel et al. 2022 compared full fine-tuning + GU with standard full fine-tuning, which is different from our work. We evaluate the cross-lingual transfer setup, which inherently comes with distribution shifts.

[7] We calculate $\mathrm{tr}(F)$ every 100 steps for all datasets (except every 50 steps for XCOPA due to its small training size).

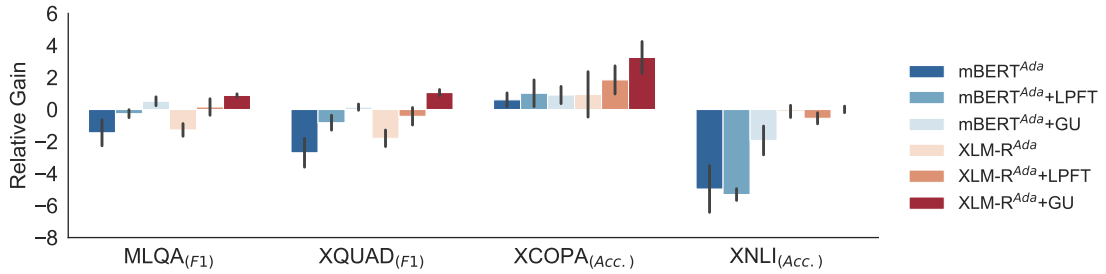
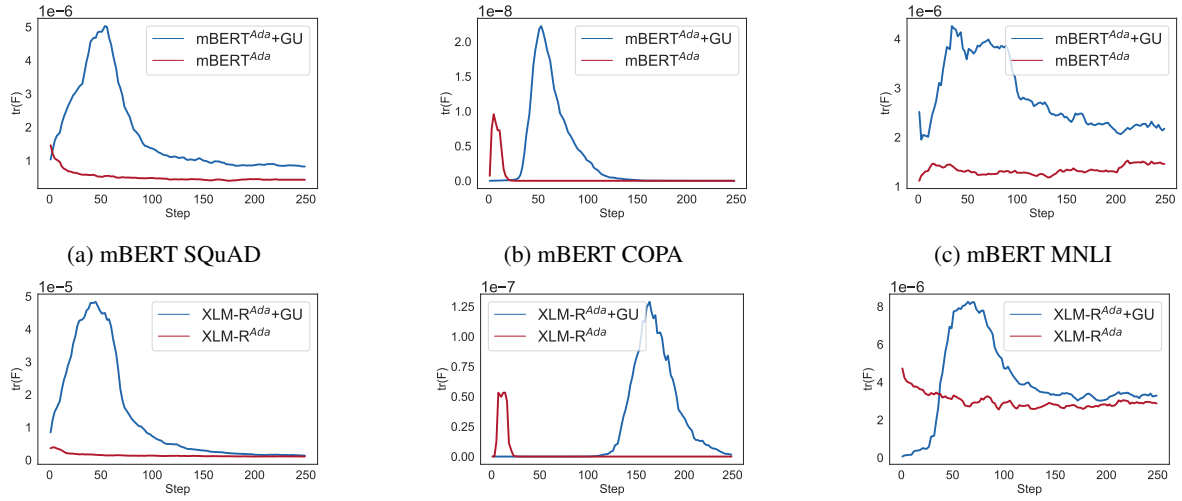


Figure 2: The relative performance of adapters fine-tuned with scheduled unfreezing (i.e., GU-based and LPFT-based task adapters) and standard fine-tuned task adapters with full fine-tuning of mBERT and XLM-R.



(a) mBERT SQuAD (b) mBERT COPA (c) mBERT MNLI

(d) XLM-R SQuAD (e) XLM-R COPA (f) XLM-R MNLI

Figure 3: Average $\mathrm{tr}(F)$ per adapter during standard training versus using gradual unfreezing.

We randomly sampled 9 schedules (those are effectively permutations of layer indices, where we also treat the standard top-down order as the 10th schedule) that start unfreezing at either layer 10 or 9, and the rest of the experimental conditions are identical.[8] We then aggregate runs with similar cross-lingual transfer results.[9]

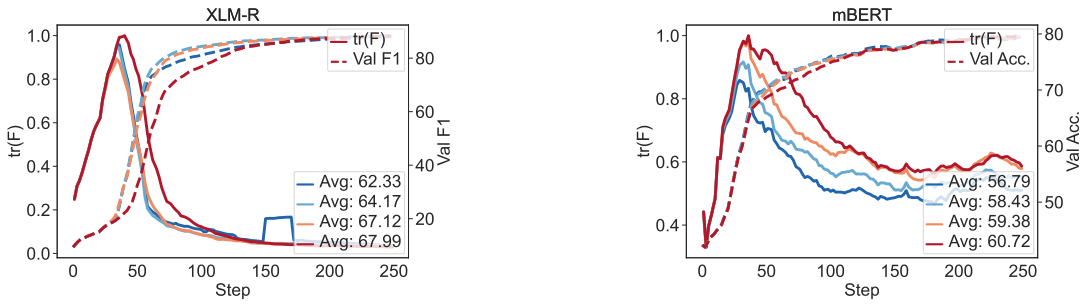
We plot the $\mathrm{tr}(F)$ along with the validation metrics in Figure 4. We observe that decreases in $\mathrm{tr}(F)$ from the peak value are associated with rapid generalization (cf., a drastic increase of the validation metrics) of the network. Previous work points out that the peak of $\mathrm{tr}(F)$ – with contradicting claims from different works (Golatkar et al., 2019; Achille et al., 2019; Jastrzebski et al., 2021) – correlates or anti-correlates with generalization, which indicates the underlining relationship may be more complex than just the peak value of the $\mathrm{tr}(F)$ curve.

Indeed, Figure 4 shows that a *wider* $\mathrm{tr}(F)$ curve (a longer learning period) often accompanies a large peak $\mathrm{tr}(F)$ value during the early phase of learning, and leads to better cross-lingual generalization during the test time. This could potentially lead to an additional new avenue of manipulating optimization with a regularization loss for cross-lingual transfer, which we leave for future work.

To the best of our knowledge, this is the first evidence indicating that $\mathrm{tr}(F)$ correlates with cross-lingual transfer performance in parameter-efficient fine-tuning and text-based Transformers. These results suggest that early-phase learning dynamics affect the generalization cross-lingually later on, and $\mathrm{tr}(F)$ can be a potential measurement to study cross-lingual generalization. From the above results, we conjecture that *inducing large* $\mathrm{tr}(F)$ *and a longer learning period would help with general-*

[8] We consider these are top layers, which likely carry similar information, in contrast to lower layers such as 0 or 1. The 11th layer is always unfrozen at the beginning, and we reject permutations of layer indices for those not starting with 10 or 9. An accepted sampled schedule could look like [10, 1, 0, 3, 9, 6, 8, 2, 5, 7, 4].

[9] This aggregation is done by: (i) sorting the cross-lingual transfer results, then (ii) taking the largest 'ungrouped' value, (iii) finding runs that are within 1 point of performance delta, and then (iv) grouping them.



(a) SQuAD (b) MNLI

Figure 4: Average $\mathrm{tr}(F)$ per adapter (normalized between 0-1 to plot together with the validation curve) and validation F1 or accuracy using a randomly sampled schedule. The average results indicated in the legend are the *averaged cross-lingual* transfer results. a) averaged F1 of MLQA and XQuAD, b) XNLI.

ization in the CF-free setting,[10] which motivates our experiments in the next section.

5.3 Auto-Scheduling by $\mathrm{tr}(F)$

We have observed that scheduled unfreezing effectively changes the learning dynamics of the task adapters. A natural follow-up question is: *If freezing certain parameters changes the learning dynamics, can we systematically and automatically select which task adapter to unfreeze next?*

To answer this question, we propose to select the next layer to unfreeze based on ranked $\mathrm{tr}(F)$ during training (i.e., Figure 1c). According to our hypothesis, an induced large and wide $\mathrm{tr}(F)$ during learning leads to better generalization (discussed in Section 5.2).[11]

$\mathrm{tr}(F)$-based Scheduled Unfreezing (FUN) Recovers GU. Surprisingly, the $\mathrm{tr}(F)$-based schedules recover the transfer performance as well as the top-down heuristic schedule (i.e., GU) in many cases. To illustrate this, we plot the unfreezing schedules generated by FUN along with the top-down schedule of GU (diagonal line) for all our experiments in Figure 5. From the figure, we can see that FUN nearly perfectly recovers the top-down schedule of GU for mBERT. We conjecture that GU is implicitly maximizing $\mathrm{tr}(F)$ at every unfreezing step. The XLM-R-based experiments largely follow the top-down order with more variance, which is likely due to noise in $\mathrm{tr}(F)$ estimation.

We show the results across all datasets in Table 2, and we include an additional baseline (+*Rand*) that randomly selects the next layer to unfreeze at every time interval k (see Appendix E for detailed results). Table 2 shows that the FUN scheduler achieves near-identical results as GU with the mBERT model, which matches the observations in Figure 5. We also achieve comparable results as GU with XLM-R. The results are well above the random unfreezing baselines and standard training (e.g., improving mBERT from standard training on XNLI for 3.58 points, or the average of 2.03 points across all 4 datasets etc.). Although the source English performance is not the focus of our work, we also find that FUN achieves better English results in most of the cases (e.g., XLM-R with FUN is better than GU in English for 6 out of 8 cases, which shows the potential of FUN beyond the context of cross-lingual transfer explored here).

In addition, we highlight that scheduled unfreezing improved transfer results for the lowest-performing language across the board. For example, FUN improved the lowest-performing language under standard adapter training in all 4 cases for the mBERT model, and in 3 out of 4 cases for XLM-R. Some gains are quite substantial, e.g., Thai with mBERT increased from 34.53 to 42.55 (see Table 2, the *Lowest* column for XQuAD).

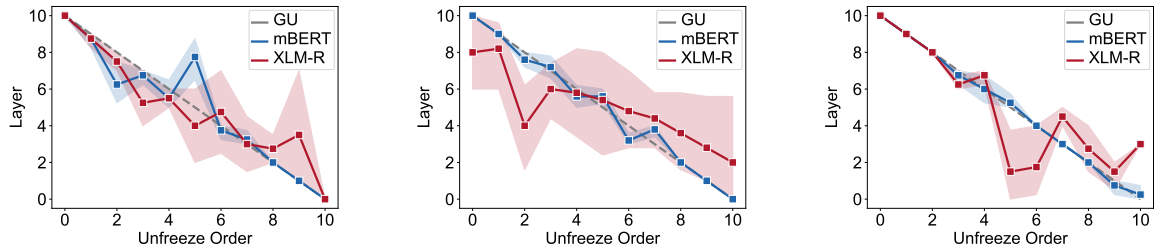
Future Perspectives of FUN. While GU remains effective, FUN offers the opportunity to potentially break away from the strict top-down unfreezing schedule and experiment with networks that have

[10] Although we empirically observe similar patterns as the previous work (Jastrzebski et al., 2021), we interpret the results differently here. Prior work argues that a sub-optimal small learning rate induces a higher peak $\mathrm{tr}(F)$ during learning, and regularizing $\mathrm{tr}(F)$ to a smaller value helps with learning and generalization. We want to point out that we experiment in different settings and look into cross-lingual generalization, which is **shifted distribution** at test time. We also use a different optimizer, large learning rates and work in a CF-free setting.

[11] $\mathrm{tr}(F)$ is calculated every k steps for L times, on 40 batches (maximum) of training data. We consider the additional computational cost to be negligible compared to GU.

	MLQA (F1 / EM)			*XQuAD (F1 / EM)*		
Method	En	Lowest (Ar)	**Average**	En	Lowest (Th)	**Average**
mBERT^{Ada}	78.99/65.85	45.76/28.77	55.40±0.94 / 37.07±0.72	83.58/71.74	34.53/38.89	60.63±1.04 / 43.90±0.85
mBERT^{Ada}+Rand	79.22/65.94	46.65/29.84	55.93±0.21 / 37.54±0.31	83.86/72.31	38.91/38.72	61.68±0.33 / 47.42±0.55
mBERT^{Ada}+GU	78.04/64.20	47.96/29.30	**57.37**±0.32 / 38.27±0.27	83.21/71.55	44.08/35.46	**63.48**±0.22 / 46.76±0.44
mBERT^{Ada}+FUN	78.82/65.29	48.20/30.90	57.33±0.51 / **38.29**±0.63	83.71/71.83	42.55/42.84	63.25±0.26 / **49.09**±0.48
Method	En	Lowest (Ar)	**Average**	En	Lowest (Ar)	**Average**
XLM-R^{Ada}	79.52/65.99	51.74/33.33	61.31±0.46 / 42.10±0.42	83.48/72.69	65.47/48.89	70.09±0.60 / 53.77±0.40
XLM-R^{Ada}+Rand	80.32/67.01	50.33/36.29	61.36±1.69 / 41.59±1.96	84.76/73.74	63.69/46.30	69.99±1.47 / 52.06±2.04
XLM-R^{Ada}+GU	80.37/66.77	55.16/35.49	**63.47**±0.12 / **43.55**±0.11	84.49/73.57	67.83/50.80	**73.04**±0.22 / **55.93**±0.15
XLM-R^{Ada}+FUN	80.92/66.70	53.17/37.73	63.10±0.79 / 43.37±0.51	84.91/73.80	66.69/49.24	72.34±0.40 / 55.21±0.63
	XCOPA (Accuracy)			*XNLI (Accuracy)*		
Method	En	Lowest (It)	**Average**	En	Lowest (Sw)	**Average**
mBERT^{Ada}	63.80	50.16	53.99±0.49	82.05	37.45	57.78±1.68
mBERT^{Ada}+Rand	65.00	50.96	53.84±0.71	81.64	45.95	59.87±0.96
mBERT^{Ada}+GU	66.60	50.00	**54.29**±0.60	81.79	54.06	**61.67**±1.04
mBERT^{Ada}+FUN	66.40	50.68	53.98±0.64	81.70	53.73	61.36±0.51
Method	En	Lowest (Ht)	**Average**	En	Lowest (Sw)	**Average**
XLM-R^{Ada}	65.20	51.28	55.93±1.58	84.31	68.16	73.31±0.44
XLM-R^{Ada}+Rand	67.20	52.12	57.05±0.42	84.52	67.29	72.68±0.56
XLM-R^{Ada}+GU	66.00	52.52	**58.24**±1.11	84.24	68.24	**73.44**±0.24
XLM-R^{Ada}+FUN	67.80	52.08	58.11±0.94	84.72	67.48	73.13±0.53

Table 2: Zero-shot transfer results across four datasets: MLQA, XQuAD, XCOPA and XNLI. Average is the cross-lingual average without English. We bold the highest and underline the second-highest average value. *Lowest* denotes the task performance for the lowest-performing target language per each evaluation dataset and base model.



(a) SQuAD (b) COPA (c) MNLI

Figure 5: Averaged unfreezing schedules for GU and FUN with different base models.

parallel structures (e.g., multiple adapters from the same depth, dual-network structures) in future work. Nonetheless, as the main finding in this work, FUN (i) provides evidence for a theory-based justification of heuristic unfreezing schedules from prior work, and (ii) it leads us to extend our understanding of learning dynamics during fine-tuning with scheduled unfreezing.

6 Conclusion

In this work, we first investigated whether scheduled unfreezing algorithms can help with generalization in the zero-shot cross-lingual transfer setting, and close the gap between parameter-efficient task-adapter training and full fine-tuning. Our experiments showed that scheduled unfreezing was indeed successful in closing this gap. Next, we investigated the training dynamics of scheduled unfreezing using the trace of the Fisher Information Matrix ($\mathrm{tr}(F)$). Our experiments revealed a link between scheduled unfreezing, $\mathrm{tr}(F)$ and generalization performance. Finally, we proposed a general scheduled unfreezing algorithm ($\mathrm{tr}(F)$-based scheduled unfreezing, FUN) that achieves performance comparable to existing heuristic variants, across multiple models and datasets. FUN enables us to gather evidence for a theory-based justification of heuristic unfreezing schedules from prior work. As FUN has more potential advantages compared to prior work, we hope to look into utilizing it for improving cross-lingual generalization capabilities of large language models in the future.

7 Limitations

In this paper, we work with the trace of the Fisher Information Matrix as the metric for studying learning dynamics. While we believe our experiments and conclusions are widely applicable, there may be other complex factors and theoretical metrics (such as the eigenvalue spectrum of F or other matrix norms, etc.) we could potentially investigate. Furthermore, our use of $\mathrm{tr}(F)$ is connected to prior work on the "critical learning period" during the early stages of training neural networks (Achille et al., 2019; Kleinman et al., 2022), which could help us gain deeper theoretical insights into parameter-efficient training methods. We also speculate GU may not degrade the performance of full parameter fine-tuning (Raffel et al., 2022) if it is done early in the training (i.e., $kL \ll N$) rather than evenly unfrozen throughout the entire training process ($k = N/L$). However, such investigations are outside of the scope of this work, and we leave it for future work.

References

Alessandro Achille, Matteo Rovere, and Stefano Soatto. 2019. Critical learning periods in deep networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Shun-ichi Amari. 1998. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276.

Alan Ansell, Edoardo Maria Ponti, Jonas Pfeiffer, Sebastian Ruder, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2021. MAD-G: Multilingual adapter generation for efficient cross-lingual transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4762–4781, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

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.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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.

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.

Rory A. Fisher. 1925. Theory of statistical estimation. *Mathematical Proceedings of the Cambridge Philosophical Society*, 22:700 – 725.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. 2019. Time matters in regularizing deep networks: Weight decay and data augmentation affect early learning dynamics, matter little near convergence. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 10677–10687. Curran Associates, Inc.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Online, Austria, May 3-7, 2021*.

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.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, pages 4411–4421.

Stanislaw Jastrzebski, Devansh Arpit, Oliver Astrand, Giancarlo B Kerg, Huan Wang, Caiming Xiong, Richard Socher, Kyunghyun Cho, and Krzysztof J Geras. 2021. Catastrophic fisher explosion: Early phase fisher matrix impacts generalization. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4772–4784. PMLR.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Michael Kleinman, Alessandro Achille, and Stefano Soatto. 2022. Critical learning periods for multisensory integration in deep networks. *arXiv preprint*, abs/2210.04643.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. 2022. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *10th International Conference on Learning Representations, ICLR 2019, Online, Apr 25-29, 2022*.

Frederik Kunstner, Philipp Hennig, and Lukas Balles. 2019. Limitations of the empirical fisher approximation for natural gradient descent. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4158–4169. Curran Associates, Inc.

Yoonho Lee, Annie S. Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. 2022. Surgical fine-tuning improves adaptation to distribution shifts. *arXiv preprint*.

Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2020. MLQA: Evaluating cross-lingual extractive question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7315–7330, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

James Martens. 2020. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press.

Marinela Parović, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2022. BAD-X: Bilingual adapters improve zero-shot cross-lingual transfer. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1791–1799, Seattle, United States. Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2021. UNKs everywhere: Adapting multilingual language models to new scripts. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10186–10203, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Edoardo Maria Ponti, Goran Glavaš, Olga Majewska, Qianchu Liu, Ivan Vulić, and Anna Korhonen. 2020. XCOPA: A multilingual dataset for causal commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2362–2376, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2022. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(1).

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.

Melissa Roemmele, Cosmin A. Bejan, and Andrew S. Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, Stanford University.

Amari Shun-ichi and Nagaoka Hiroshi. 2000. Methods of information geometry. volume 191 of *Translations of Mathematical Monographs*.

Sidak Pal Singh and Dan Alistarh. 2020. Woodfisher: Efficient second-order approximation for neural network compression. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, Online*. Curran Associates, Inc.

Asa Cooper Stickland and Iain Murray. 2019. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995. PMLR.

Yi-Lin Sung, Varun Nair, and Colin Raffel. 2021. Training neural networks with fixed sparse masks. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, Online*, pages 24193–24205. Curran Associates, Inc.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online. Association for Computational Linguistics.

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.

Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. 2021. Raise a child in large language model: Towards effective and generalizable fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9514–9528, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhuoyi Yang, Ming Ding, Yanhui Guo, Qingsong Lv, and Jie Tang. 2022. Parameter-efficient tuning makes a good classification head. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dahbi, United Arab Emirates. Association for Computational Linguistics.

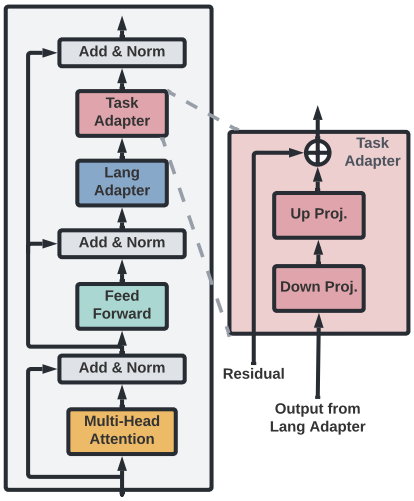


Figure 6: Adapter Architecture.

A Adapter Architecture

Figure 6 shows the adapter architecture for cross-lingual transfer based on MAD-X.

B Hyperparameters

We include the detailed hyperparameters used in our experiments in Table 3 and 4. We use the AdamW optimizer (Loshchilov and Hutter, 2019) for all experiments, without warmups.

We use a constant learning rate scheduler and a maximum gradient norm of 1 in all models for training with SQuAD and COPA.

For COPA we train the model with longer epochs for scheduled unfreezing. since the training data for COPA only contains 400 instances, the training time is still very small (<1 hour). We found standard training for task adapters for COPA does not benefit from longer training time and a smaller learning rate.

In addition, some of the language adapters are missing from the AdapterHub (just for mBERT or both for mBERT and XLM-R). We follow the configuration from (Pfeiffer et al., 2020), and train those language adapters that are missing for mBERT (Italian, Tamil and Thai) with the Wikipedia data, learning rate of 1e-4 , batch size of 64, sequence length of 512, and a maximum 100 epochs/training budget or 24 hours (whichever is reached first).

The task adapters have a reduction factor of 16 as indicated in MAD-X.

Model	Epochs	lr	batchsize	k-LPFT	k-GU	k-FUN
			SQuAD			
mBERT	5	5e-4	32	800	800	800
XLM-R	15	2e-4/5e-4	32	800	800	100
			COPA			
mBERT	500/5000	1e-4/1e-5	64	800	50	50
XLM-R	500/5000	1e-4/1e-5	64	800	1000	1000
			MNLI			
mBERT	15	5e-4	128	25	800	50
XLM-R	15	5e-4	128	800	800	800

Table 3: Hyperparameters used in the main experiments. x/y denotes: x for standard adapter training and y for all scheduled unfreezing experiments.

Model	Epochs	lr	batchsize	k-1k	k-5k	k-10k
			SQuAD			
mBERT	20	5e-4	32	10	50	50
XLM-R	20	5e-4	32	10	50	50
			MNLI			
mBERT	50	5e-4	128	1	25	25
XLM-R	50	5e-4	128	1	25	10

Table 4: Hyperparameters used in the experiments with reduced task training data.

C Dataset Statistics

We include the dataset statistics in Table 5. The training data for XQuAD and MLQA are SQuAD (Rajpurkar et al., 2016). The training data for XCOPA is COPA (Roemmele et al., 2011) and the training data for XNLI is MNLI (Williams et al., 2018). All datasets used are available on HuggingFace. The language names and codes in our experiments are in Table 6.

Train data / Test data	n. train (En)	n. val (En)	n. test	n. lang
SQuAD / XQuAD	87599	10570	1190	11
SQuAD / MLQA	87599	10570	4517 – 5495	5
COPA / XCOPA	400	100	500	11
MNLI / XNLI	392702	2490	5010	14

Table 5: Dataset statistics.

D Pseudo Code for $\mathrm{tr}(F)$ calculation

We provide the pseudo code for $\mathrm{tr}(F)$ calculation in Algorithm 2. Alternatively, please see our code at URL. Let FORWARD$(*)$ be the standard forward pass operation, SAMPLE$(*)$ be a function sampling labels from the label distribution of

Language	Code	Language	Code	Language	Code
Arabic	Ar	German	De	Greek	El
Spanish	Es	Hindi	Hi	Russian	Ru
Thai	Th	Turkish	Tr	Vietnamese	Vi
Estonian	Et	Haitian	Ht	Italian	It
Indonesian	Id	Quechua	Qu	Swahili	Sw
Chinese	Zh	Tamil	Ta		

Table 6: Language code.

Model	MLQA (F1)	XQuAD (F1)
mBERT	56.85	63.33
XLM-R	62.59	71.98
Model	**XCOPA (Acc.)**	**XNLI (Acc.)**
mBERT	53.39	63.60
XLM-R	54.99	73.43

Table 7: Baselines (full fine-tuning) results for cross-lingual transfers.

data, and AGGAVG$(*)$ the function that aggregates $\mathrm{tr}(F)$ by task adapter blocks then taking the average over the number of trainable layers.

Algorithm 2 $\mathrm{tr}(F)$ Calculation

Require: Number of batches N to sample from training data D for computing $\mathrm{tr}(F)$, $\mathcal{P}$ trainable parameters.

```
Copy the model.                ▷ To avoid interfering standard optimization.
for i = 1...N do
    Sample a data batch b ~ D
    outputs = FORWARD(*)
    labels = SAMPLE(*)         ▷ From the possible y from the dataset.
    prob = LogSoftmax(outputs)
    loss = NLL(prob; labels)
    loss.backward()
    for p_j = 1...|P| do
        tr(F)_j = p_j.grad^2 / |b|
    end for
    tr(F) = AGGAVG(*)
end for
```

E Detailed Results for Experiments

We show the detailed per-language experimental results in Tables 8 to 10.

The baselines (full parameter fine-tuning) results for plotting Figure 2 are in Table 7.

F Additional Experiments

F.1 Reverse Gradual Unfreezing

We briefly experimented (2 runs) with the reverse order (bottom-up) of gradual unfreezing on MLQA, XQuAD and XNLI. We include the results for bottom-up GU $(+(rev))$ in Figure 11, and included the numbers from standard GU for reference. The cross-lingual transfer results are significantly lower than the standard GU; however, the English results are similar.

F.2 Experiments on Smaller Task Training Data with $\mathrm{tr}(F)$-based Scheduling

We additionally performed the same experiments with smaller training data as described in our main paper, but now with $\mathrm{tr}(F)$-based scheduling $(+FUN)$. The results are in Table 12. Our results show that FUN is also comparable to GU when there are fewer training instances available. The k used in our experiment for FUN are (for 1k/5k/10k correspondingly): mBERT-XQuAD = [10,50,50], XLM-R-XQuAD = [10,50,25], mBERT-XNLI = [10,50,25], XLM-R-XNLI = [1,25,10]. The remaining hyperparameters are the same as in all other experiments.

F.3 Preliminary Results with mDeBERTa

We include additional experiments on another, more recent base model, mDeBERTa (He et al., 2021). mDeBERTa is the multilingual version of the recently proposed DeBERTa (He et al., 2021) model with disentangled attention to its word content and position representations. We used the (mdeberta-v3-base) model for all our experiments.

Note that we trained language adapters using MLM loss for the mDeBERTa model according to the setup described in (Pfeiffer et al., 2020). However, we see very large discrepancies in terms of transfer results for both XCOPA and XNLI when compared to the standard fine-tuning (the gaps are also much larger than the gaps for mBERT or XLM-R). We hypothesize that the discrepancies may be because mDeBERTa uses different attentions and the adapters we studied here are not designed for mDeBERTa (both in their architecture and in their training method). However, as tuning adapter architectures is beyond the scope of our study, we include the results here for completeness only.

MLQA	En (F1 / EM)	Ar	De	El	Es	Hi	Ru	Th	Tr	Vi	Zh	**Avg. F1 / EM**
mBERT^{Ada}	78.99/65.85	45.76	59.47	-	64.60	46.91	-	-	-	57.01	58.64	55.40±0.94 / 37.07±0.72
mBERT^{Ada}+Rand	79.22/65.94	46.65	57.92	-	67.91	46.03	-	-	-	57.75	59.34	55.93±0.21 / 37.54±0.31
mBERT^{Ada}+GU	78.04/64.20	47.96	57.95	-	68.64	51.75	-	-	-	58.95	58.95	57.37±0.32 / 38.27±0.27
mBERT^{Ada}+FUN	78.82/65.29	48.20	58.96	-	67.19	51.51	-	-	-	59.47	58.64	57.33±0.51 / 38.29±0.63
XLM-R^{Ada}	79.52/65.99	51.74	59.64	-	68.33	61.77	-	-	-	64.85	61.88	61.31±0.46 / 42.10±0.42
XLM-R^{Ada}+Rand	80.32/67.01	50.33	61.72	-	69.98	60.08	-	-	-	63.81	62.22	61.36±1.69 / 41.59±1.96
XLM-R^{Ada}+GU	80.37/66.77	55.16	61.30	-	70.36	63.75	-	-	-	66.45	63.79	63.47±0.12 / 43.55±0.11
XLM-R^{Ada}+FUN	80.92/66.70	53.17	62.38	-	70.04	63.77	-	-	-	65.38	63.86	63.10±0.79 / 43.37±0.51
XQuAD	En (F1 / EM)	Ar	De	El	Es	Hi	Ru	Th	Tr	Vi	Zh	**Avg. F1 / EM**
mBERT^{Ada}	83.58/71.74	57.95	71.20	57.60	73.11	53.75	70.05	34.53	50.15	68.38	69.56	60.63±1.04 / 43.90±0.85
mBERT^{Ada}+Rand	83.86/72.31	59.09	71.58	62.06	74.84	52.61	70.00	38.91	48.49	69.29	69.92	61.68±0.33 / 47.42±0.55
mBERT^{Ada}+GU	83.21/71.55	62.90	72.39	62.38	74.43	56.28	69.46	44.08	53.39	70.10	69.37	63.48±0.22 / 46.76±0.44
mBERT^{Ada}+FUN	83.71/71.83	62.24	72.74	62.19	74.05	56.92	69.74	42.55	53.40	69.56	69.13	63.25±0.26 / 49.09±0.48
XLM-R^{Ada}	83.48/72.69	65.47	72.74	71.92	74.88	67.70	73.58	66.53	66.36	72.38	69.36	70.09±0.60 / 53.77±0.40
XLM-R^{Ada}+Rand	84.76/73.74	63.69	74.40	71.25	76.28	65.09	73.77	64.93	64.85	72.06	73.54	69.99±1.47 / 52.06±2.04
XLM-R^{Ada}+GU	84.49/73.57	67.83	75.55	74.26	77.42	70.46	75.52	69.52	68.53	75.88	75.39	73.04±0.22 / 55.93±0.15
XLM-R^{Ada}+FUN	84.91/73.80	66.69	75.94	74.07	76.58	69.59	75.48	67.59	68.19	74.52	74.77	72.34±0.40 / 55.21±0.63

Table 8: Zero-shot transfer results (F1) for MLQA and XQuAD. Average is the cross-lingual average without English.

XCOPA	En	Et	Ht	It	Id	Qu	Sw	Zh	Ta	Th	Tr	Vi	**Avg. Acc.**
mBERT^{Ada}	63.80	54.20	53.04	50.16	53.84	53.12	54.16	59.08	52.56	51.68	54.52	57.56	53.99±0.49
mBERT^{Ada}+Rand	65.00	53.36	52.32	50.96	53.60	54.00	53.44	58.64	50.92	51.96	55.04	57.96	53.84±0.71
mBERT^{Ada}+GU	66.60	54.44	52.60	50.00	54.88	53.52	53.52	59.76	52.12	52.36	55.44	58.60	54.29±0.60
mBERT^{Ada}+FUN	66.40	53.44	52.92	50.68	54.76	53.48	54.40	59.00	51.36	51.08	54.32	58.36	53.98±0.64
XLM-R^{Ada}	65.20	56.16	51.28	56.72	58.00	51.80	55.60	59.12	56.44	57.72	56.48	55.96	55.93±1.58
XLM-R^{Ada}+Rand	67.20	57.08	52.12	57.80	60.72	53.32	56.24	60.08	57.36	58.20	56.76	57.88	57.05±0.42
XLM-R^{Ada}+GU	66.00	58.56	52.52	58.24	62.04	53.96	56.88	61.36	59.00	60.08	58.52	59.52	58.24±1.11
XLM-R^{Ada}+FUN	67.80	58.16	52.08	57.44	61.28	55.04	56.36	61.64	57.84	61.04	58.80	59.48	58.11±0.94

Table 9: Zero-shot transfer results (Accuracy) for XCOPA. Average is the cross-lingual average without English.

XNLI	En	Ar	De	El	Es	Hi	Ru	Sw	Th	Tr	Vi	Zh	**Avg. Acc.**
mBERT^{Ada}	82.05	42.09	65.81	62.16	70.84	57.92	63.76	37.45	40.89	61.53	68.01	65.08	57.78±1.68
mBERT^{Ada}+Rand	81.64	53.98	66.32	62.85	70.33	58.15	65.08	45.95	41.80	61.25	67.73	65.12	59.87±0.96
mBERT^{Ada}+GU	81.79	62.78	66.25	63.51	70.28	58.89	65.74	54.06	38.97	62.25	68.52	67.17	61.67±1.04
mBERT^{Ada}+FUN	81.70	58.48	66.32	63.87	70.98	59.08	65.45	53.73	41.13	61.89	68.25	65.75	61.36±0.51
XLM-R^{Ada}	84.31	70.42	76.16	75.80	78.85	70.16	75.14	68.16	71.14	72.33	75.06	73.24	73.31±0.44
XLM-R^{Ada}+Rand	84.52	69.91	75.91	75.05	78.04	69.56	74.51	67.29	70.40	72.05	74.25	72.48	72.68±0.56
XLM-R^{Ada}+GU	84.24	70.22	75.92	75.7	78.32	70.61	75.70	68.24	71.99	71.97	75.36	73.82	73.44±0.24
XLM-R^{Ada}+FUN	84.72	70.58	76.17	75.68	78.29	69.75	75.42	67.48	71.44	71.71	74.75	73.20	73.13±0.53

Table 10: Zero-shot transfer results (Accuracy) for XNLI. Average is the cross-lingual average without English.

MLQA	En (F1)	**Avg. F1**
mBERT^{Ada}+GU	78.04	57.37
mBERT^{Ada}+GU (rev)	78.71	49.09
XLM-R^{Ada}+GU	80.37	63.47
XLM-R^{Ada}+GU (rev)	81.38	57.59
XQuAD	En (F1)	**Avg. F1**
mBERT^{Ada}+GU	83.21	63.48
mBERT^{Ada}+GU (rev)	82.17	53.43
XLM-R^{Ada}+GU	84.49	73.04
XLM-R^{Ada}+GU (rev)	84.12	65.44
XNLI	En (Acc.)	**Avg. Acc.**
mBERT^{Ada}+GU	81.79	61.67
mBERT^{Ada}+GU	81.43	55.67
XLM-R^{Ada}+GU	84.24	73.44
XLM-R^{Ada}+GU (rev)	84.23	72.50

Table 11: Zero-shot transfer results of gradual unfreezing in reverse order across three datasets: MLQA, XQuAD, and XNLI. Average is the cross-lingual average without English.

XQuAD ((F1))	1K	5K	10K
mBERT^{Ada}	45.27±0.59	52.58±0.81	55.89±1.08
mBERT^{Ada}+GU	45.93±0.50	53.10±0.35	56.47±0.74
mBERT^{Ada}+FUN	46.30±0.71	53.68±0.38	56.50±0.97
XLM-R^{Ada}	44.11±1.43	57.20±0.36	61.75±0.68
XLM-R^{Ada}+GU	48.42±1.20	59.88±1.51	65.28±0.76
XLM-R^{Ada}+FUN	47.67±1.73	60.72±1.07	65.16±1.16
XNLI (Accuracy)	1K	5K	10K
mBERT^{Ada}	43.86±1.43	49.68±0.73	52.34±0.40
mBERT^{Ada}+GU	44.69±0.61	51.67±0.43	53.95±1.47
mBERT^{Ada}+FUN	44.86±0.49	51.61±0.58	53.40±0.93
XLM-R^{Ada}	52.75±2.03	64.22±1.04	65.80±0.61
XLM-R^{Ada}+GU	52.86±1.38	64.15±0.35	65.91±0.56
XLM-R^{Ada}+FUN	52.29±1.85	64.10±0.92	66.26±0.87

Table 12: Cross-lingual transfer performance with subsampled English task data for task fine-tuning.

XQuAD	En (F1 / EM)	Ar	De	El	Es	Hi	Ru	Th	Tr	Vi	Zh	**Avg. F1 / EM**
mDeBERTa^{Ada}	82.31	58.50	67.56	-	73.00	64.27	-	-	-	64.92	65.27	65.59±0.22 / 46.19±0.29
mDeBERTa^{Ada}+GU	82.27	61.19	67.01	-	73.26	65.71	-	-	-	65.71	67.44	67.08±0.38 / 47.22±0.32
MLQA	En (F1 / EM)	Ar	De	El	Es	Hi	Ru	Th	Tr	Vi	Zh	**Avg. F1 / EM**
mDeBERTa^{Ada}	86.30	72.33	80.38	76.89	79.93	72.77	77.27	69.97	71.51	74.99	78.98	75.50±0.29 / 58.66±0.19
mDeBERTa^{Ada}+GU	85.52	73.31	79.59	78.41	79.77	73.58	77.87	70.98	72.66	75.42	79.15	76.07±0.13 / 58.90±0.12

Table 13: mDeBERTa: Zero-shot transfer results (F1) XQUAD and MLQA. Average is the cross-lingual average without English.

XCOPA	En	Et	Ht	It	Id	Qu	Sw	Zh	Ta	Th	Tr	Vi	**Avg. Acc.**
mDeBERTa^{Ada}	65.75	55.32	55.68	58.64	61.00	51.40	56.48	62.76	57.16	57.28	55.76	59.04	57.32±4.46
mDeBERTa^{Ada}+GU	65.80	57.96	56.60	59.36	60.12	52.36	55.96	63.28	57.88	57.48	58.12	58.76	57.99±3.57

Table 14: mDeBERTa: Zero-shot transfer results (Accuracy) XCOPA. Average is the cross-lingual average without English.

XNLI	En	Ar	De	El	Es	Hi	Ru	Sw	Th	Tr	Vi	Zh	**Avg. Acc.**
mDeBERTa^{Ada}	86.94	72.09	78.57	76.03	80.92	69.41	76.98	68.45	68.73	75.93	74.62	73.30	74.09±0.77
mDeBERTa^{Ada}+GU	86.48	71.46	77.90	75.50	79.23	67.30	75.84	68.64	69.68	74.51	74.05	74.85	73.54±0.58

Table 15: mDeBERTa: Zero-shot transfer results (Accuracy) XNLI. Average is the cross-lingual average without English.