# SEQUENTIAL REPTILE: INTER-TASK GRADIENT ALIGNMENT FOR MULTILINGUAL LEARNING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Multilingual models jointly pretrained on multiple languages have achieved remarkable performance on various multilingual downstream tasks. Moreover, models finetuned on a single monolingual downstream task have shown to generalize to unseen languages. In this paper, we first show that it is crucial for those tasks to align gradients between them in order to maximize knowledge transfer while minimizing negative transfer. Despite its importance, the existing methods for gradient alignment either have a completely different purpose, ignore inter-task alignment, or aim to solve continual learning problems in rather inefficient ways. As a result of the misaligned gradients between tasks, the model suffers from severe negative transfer in the form of catastrophic forgetting of the knowledge acquired from the pretraining. To overcome the limitations, we propose a simple yet effective method that can efficiently align gradients between tasks. Specifically, we perform each inner-optimization by sequentially sampling batches from all the tasks, followed by a Reptile outer update. Thanks to the gradients aligned between tasks by our method, the model becomes less vulnerable to negative transfer and catastrophic forgetting. We extensively validate our method on various multi-task learning and zero-shot cross-lingual transfer tasks, where our method largely outperforms all the relevant baselines we consider.

## 1 INTRODUCTION

Multilingual language models (Devlin et al., 2019; Conneau & Lample, 2019; Conneau et al., 2020; Liu et al., 2020; Lewis et al., 2020a; Xue et al., 2021) have achieved impressive performance on a variety of multilingual natural language processing (NLP) tasks. Training a model with multiple languages jointly can be understood as a multi-task learning (MTL) problem where each language serves as a distinct task to be learned (Wang et al., 2021). The goal of MTL is to make use of relatedness between tasks to improve generalization without negative transfer (Kang et al., 2011; Kumar & Daumé III, 2012; Lee et al., 2016; Wang et al., 2019a; 2020b). Likewise, when we train with a downstream multilingual MTL objective, we need to maximize knowledge transfer between the languages while minimizing negative transfer between them. This is achieved by developing an effective MTL strategy that can prevent the model from memorizing task-specific knowledge not easily transferable across the languages.

Such MTL problem is highly related to the gradient alignment between the tasks, especially when we finetune a well-pretrained model like multilingual BERT (Devlin et al., 2019). We see from the bottom path of Fig. 1 that the cosine similarity between the task gradients (gradients of MTL losses individually computed for each task) tend to gradually decrease as we finetune the model with the MTL objective. It means that the model gradually starts memorizing task-specific (or language-specific) knowledge not compatible across the languages, which can cause a negative transfer from one language to another. In case of finetuning the well-pretrained model, we find that it causes catastrophic forgetting of the pretrained knowledge. Since the pretrained model is the fundamental knowledge shared across all NLP tasks, such catastrophic forgetting can severely degrade the performance of all tasks. Therefore, we want our model to maximally retain the knowledge of the pretrained model by finding a good trade-off between minimizing the downstream MTL loss and maximizing the cosine similarity between the task gradients, as illustrated in the upper path of Fig. 1. In this paper, we aim to solve this problem by developing an MTL method that can efficiently align gradients across the tasks while finetuning the pretrained model.

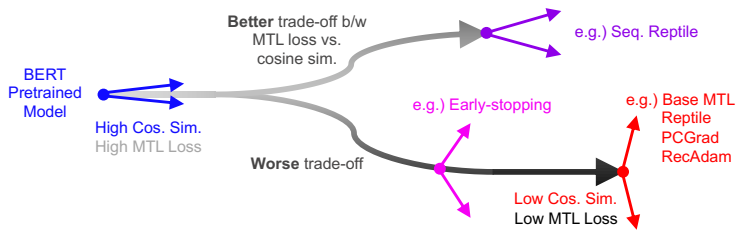



Figure 1: **Concepts.** Black arrows denote finetuning processes. The darker the part of the arrows, the lower the MTL loss. Upper and bottom path shows better and worse trade-off, respectively. Colored arrows denote task gradients. Blue and red color shows high and low cosine similarity, respectively. We demonstrate this concept with the actual experimental results in Fig. 7a.

There has been a seemingly related observation by Yu et al. (2020) that the conflict (negative cosine similarity) between task gradients makes it hard to optimize the MTL objective. They propose to manually alter the direction of task gradients whenever the task gradients conflict to each other. However, their intuition is completely different from ours. They *manually* modify the task gradients whenever the gradient conflicts happen, which leads to more aggressive optimization of MTL objective. In case of finetuning a well-pretrained model, we find that it simply leads to catastrophic forgetting. Instead, we aim to make the model converge to a point where task gradients are naturally aligned, leading to less aggressive optimization of the MTL objective (See the upper path of Fig. 1).

Then a natural question is if we can alleviate the catastrophic forgetting with early stopping. Our observation is that whereas early stopping can slightly increase cosine similarity to some extent, it is not sufficient to find a good trade-off between minimizing MTL objective and maximizing cosine similarity to improve generalization (See Fig. 1). It means that we may need either an implicit or explicit objective for gradient alignment between tasks. Also, Chen et al. (2020) recently argue that we can mitigate catastrophic forgetting by adding $\ell_2$ regularization to AdamW optimizer (Loshchilov & Hutter, 2019). They argue that the resultant optimizer penalizes $\ell_2$ distance from the pretrained model during the finetuning stage. However, unfortunately we find that their method is not much effective in preventing catastrophic forgetting in the experimental setups we consider.

On the other hand, Reptile (Nichol et al., 2018) implicitly promotes gradient alignment between mini-batches *within a task*. Reptile updates a shared initial parameter individually for each task, such that the task gradients are not necessarily aligned across the tasks. In continual learning area, MER (Riemer et al., 2019) and La-MAML (Gupta et al., 2020) propose to align the gradients between sequentially incoming tasks in order to maximally share the progress on their objectives. However, as they focus on continual learning problems, they require explicit memory buffers to store previous task examples and align gradients with them, which is complicated and costly. Further, their methods are rather inefficient in that the inner-optimization is done with batch size set to 1, which takes significantly more time than usual batch-wise training. Therefore, their methods are not straightforwardly applicable to multilingual MTL problems we aim to solve in this paper.

In this paper, we show that when we finetune a well-pretrained model, it is sufficient to align gradients between the currently given downstream tasks in order to retain the pretrained knowledge, without accessing the data used for pretraining or memory buffers. Specifically, during the finetuning stage, we sequentially sample mini-batches *from all the downstream tasks* at hand to perform a single inner-optimization, followed by a Reptile outer update. Then, we can efficiently align the gradients between tasks based on the implicit dependencies between the inner-update steps. This procedure, which we call *Sequential Reptile*, is a simple yet effective method that can largely improve the performance of various downstream multilingual tasks by preventing negative transfer and catastrophic forgetting in an efficient manner. We summarize our contributions as follows.

- We show that when finetuning a well-pretrained model like BERT, gradients not aligned between tasks can cause negative transfer and catastrophic forgetting of the knowledge acquired from the pretraining.
- To solve this problem, we propose *Sequential Reptile*, a simple yet effective MTL method that can efficiently align gradients between tasks, thereby prevent negative transfer and catastrophic forgetting.
- We extensively validate our method on various MTL and zero-shot cross-lingual transfer tasks, including question answering, named entity recognition and natural language inference tasks, in which our method largely outperforms all the baselines.

## 2 RELATED WORKS

**Multi-task Learning** The goal of multi-task learning (MTL) is to leverage relatedness between tasks for effective knowledge transfer while preventing negative interference between them (Zhang & Yeung, 2010; Kang et al., 2011; Lee et al., 2016). GradNorm (Chen et al., 2018) tackles task imbalance problem by adaptively weighting each task loss. Another line of literature propose to search Pareto optimal solutions which represent trade-offs between the tasks (Sener & Koltun, 2018; Lin et al., 2019). Recently, PCGrad (Yu et al., 2020) and GradVac (Wang et al., 2021) propose to manually resolve the conflict between task gradients to more aggressively optimize the given MTL objective. However, their goal is completely different from ours because here we focus on preventing negative transfer by finding a model that can naturally align the task gradients without such manual modifications.

**Multilingual Language Model** Training a multilingual language model is a typical example of multi-task learning. Most of the previous works focus on jointly pretraining a model with hundreds of languages to transfer common knowledge between the languages (Devlin et al., 2019; Conneau & Lample, 2019; Conneau et al., 2020; Liu et al., 2020; Lewis et al., 2020a; Xue et al., 2021). Some literature show the limitation of jointly training the model with multilingual corpora (Arivazhagan et al., 2019; Wang et al., 2020b). Several follow-up works propose to tackle the various accompanying problems such as post-hoc alignment (Wang et al., 2019b; Cao et al., 2019) and data balancing (Wang et al., 2020a). In this paper, we focus on how to finetune a well pretrained multilingual language model by preventing catastrophic forgetting of the pretrained knowledge.

**Zero-shot Cross Lingual Transfer** Zero-shot cross-lingual transfer is to train a model with monolingual labeled data and evaluate it on some unseen target languages without further finetuning the model on the target languages. Nooralahzadeh et al. (2020) utilize meta-learning to learn how to transfer knowledge from high resource languages to low resource ones. Hu et al. (2021) and Pan et al. (2021) leverage a set of paired sentences from different languages to train the model and minimize the distance between the representation of the paired sentences. Instead, we partition the monolingual labeled data into groups and consider them as a set of tasks for multi-task learning.

## 3 APPROACH

The goal of multi-task learning (MTL) is to estimate a model parameter $\phi$ that can achieve good performance across all the given $T$ tasks, where each task $t = 1, \ldots, T$ has task-specific data $\mathcal{D}_t$. We learn $\phi$ by minimizing the sum of task losses.

$$\min_{\phi} \sum_{t=1}^{T} \mathcal{L}(\phi; \mathcal{D}_t) + \lambda \Omega(\phi) \tag{1}$$

where $\Omega(\phi)$ is a regularization term and $\lambda \geq 0$ is an associated coefficient.

**Reptile** We briefly review Reptile (Nichol et al., 2018), an efficient first-order meta-learning method suitable for large-scale learning scenario. We show that Reptile has an approximate learning objective of the form in Eq. 1. Although Reptile is originally designed for learning a shared initialization, we can use the initialization $\phi$ for actual predictions without any adaptation (Riemer et al., 2019). Firstly, given a set of $T$ tasks, we *individually* perform the task-specific optimization from $\phi$. Specifically, for each task $t = 1, \ldots, T$, we perform the optimization by sampling mini-batches $\mathcal{B}_t^{(1)}, \ldots, \mathcal{B}_t^{(K)}$ from task data $\mathcal{D}_t$ and taking gradient steps with them.

$$\theta_t^{(0)} = \phi, \qquad \theta_t^{(k)} = \theta_t^{(k-1)} - \alpha \frac{\partial \mathcal{L}(\theta_t^{(k-1)}; \mathcal{B}_t^{(k)})}{\partial \theta_t^{(k-1)}} \tag{2}$$

for $k = 1, \ldots, K$, where $\alpha$ is an inner-learning rate and $\theta_t^{(k)}$ denotes the task-specific parameter of task $t$ evolved from $\phi$ by taking $k$ gradient steps. After performing $K$ gradient steps for all $T$ tasks, we meta-update $\phi$ as follows:

$$\phi \leftarrow \phi - \eta \cdot \frac{1}{T} \sum_{t=1}^{T} \mathrm{MG}_t(\phi), \quad \text{where} \quad \mathrm{MG}_t(\phi) = \phi - \theta_t^{(K)} \tag{3}$$

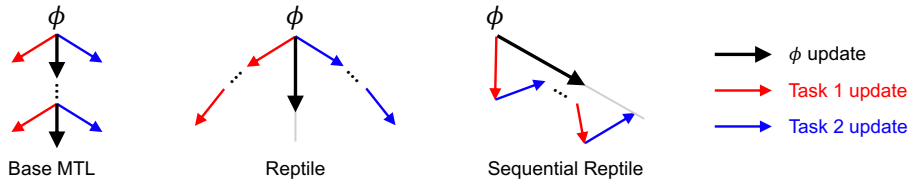



Figure 2: Comparison between the methods.

where $\eta$ denotes an outer-learning rate. Nichol et al. (2018) show that expectation of $\text{MG}_t(\phi)$ over the random sampling of batches, which is the meta-gradient of task $t$ evaluated at $\phi$, can be approximated as follows based on Taylor expansion:

$$\mathbb{E}\left[\text{MG}_t(\phi)\right] \approx \frac{\partial}{\partial \phi}\mathbb{E}\left[\sum_{k=1}^{K}\mathcal{L}(\phi;\mathcal{B}_t^{(k)}) - \frac{\alpha}{2}\sum_{k=1}^{K}\sum_{j=1}^{k-1}\left\langle \frac{\partial \mathcal{L}(\phi;\mathcal{B}_t^{(k)})}{\partial \phi}, \frac{\partial \mathcal{L}(\phi;\mathcal{B}_t^{(j)})}{\partial \phi}\right\rangle\right] \tag{4}$$

where $\langle \cdot, \cdot \rangle$ denotes a dot product. We can see that $\mathbb{E}[\text{MG}_t(\phi)]$ approximately minimizes the task-specific loss (first term) and maximizes the inner-products between gradients computed with different batches (second term). The inner-learning rate $\alpha$ controls the trade-off between them. However, the critical limitation is that the dot product does not consider aligning gradients computed from different tasks. This is because each inner-learning trajectory consists of the batches $\mathcal{B}_t^{(1)}, \ldots, \mathcal{B}_t^{(K)}$ sampled from the same task data $\mathcal{D}_t$.

### 3.1 Sequential Reptile

In order to consider gradient alignment *across tasks* as well, we propose to let the inner-learning trajectory consist of mini-batches randomly sampled *from all tasks*, which we call *Sequential Reptile*.

Unlike Reptile where we run $T$ task-specific inner-learning trajectory (in parallel), now we have a single learning trajectory responsible for all $T$ tasks (See Figure 2). Specifically, for each inner-step $k$, we randomly sample a task index $t_k \in \{1, \ldots, T\}$ and a corresponding mini-batch $\mathcal{B}_{t_k}^{(k)}$, and then sequentially update $\theta^{(k)}$ as follows.

$$\theta^{(0)} = \phi, \qquad \theta^{(k)} = \theta^{(k-1)} - \alpha \frac{\partial \mathcal{L}(\theta^{(k-1)};\mathcal{B}_{t_k}^{(k)})}{\partial \theta^{(k-1)}}, \quad \text{where} \quad t_k \sim \text{Cat}(p_1, \ldots, p_T) \tag{5}$$

$\text{Cat}(p_1, \ldots, p_T)$ is a categorical distribution parameterized by $p_1, \ldots, p_T$, the probability of each task to be selected. For example, we can let $p_t \propto (N_t)^q$ where $N_t$ is the number of training instances for task $t$ and $q$ is some constant. After $K$ gradient steps with Eq. 5, we update $\phi$ as follows

$$\phi \leftarrow \phi - \eta \cdot \text{MG}(\phi), \quad \text{where} \quad \text{MG}(\phi) = \phi - \theta^{(K)} \tag{6}$$

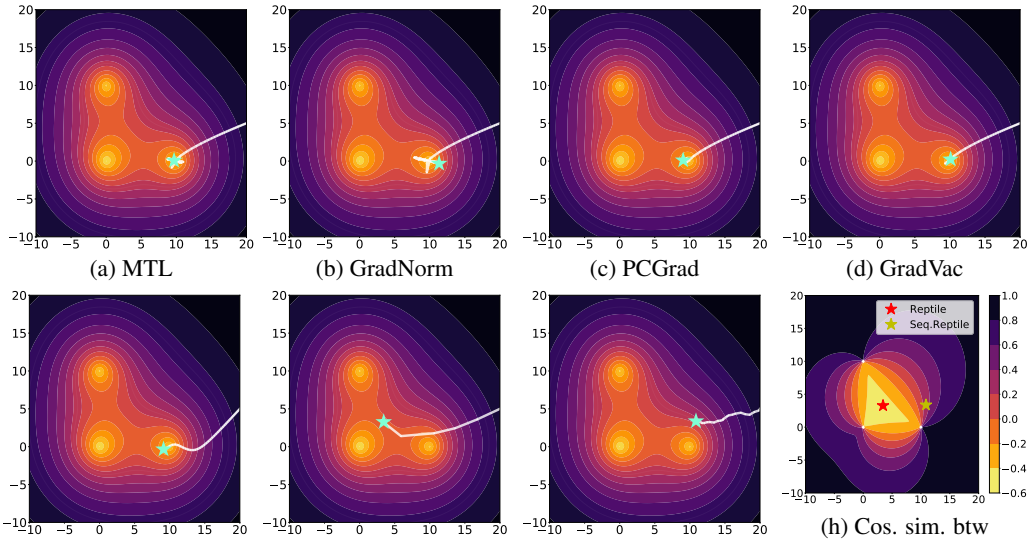
Again, based on Taylor expansion, we have the following approximate form for expectation of the meta-gradient $\text{MG}(\phi)$ over the random sampling of tasks (See derivation in Appendix C).

$$\mathbb{E}\left[\text{MG}(\phi)\right] \approx \frac{\partial}{\partial \phi}\mathbb{E}\left[\sum_{k=1}^{K}\mathcal{L}(\phi;\mathcal{B}_{t_k}^{(k)}) - \frac{\alpha}{2}\sum_{k=1}^{K}\sum_{j=1}^{k-1}\left\langle \frac{\partial \mathcal{L}(\phi;\mathcal{B}_{t_k}^{(k)})}{\partial \phi}, \frac{\partial \mathcal{L}(\phi;\mathcal{B}_{t_j}^{(j)})}{\partial \phi}\right\rangle\right] \tag{7}$$

Note that the critical difference of Eq. 7 from Eq. 4 is that the dot product between the two gradients is computed from the different tasks, $t_k$ and $t_j$. Such inter-task dependency appears as we randomly sample batches from all the tasks sequentially and compose a single learning trajectory with them. As a result, Eq. 7, or Sequential Reptile promotes the gradient alignments between the tasks, preventing the model from memorizing task or language specific knowledge. It also means that the model can find a good trade-off between minimizing the MTL objective and inter-task gradient alignment, thereby effectively preventing catastrophic forgetting of the knowledge acquired from pretraining.

## 4 Experiments

We first verify our hypothesis with synthetic experiments. We then validate our method by solving multi-task learning (MTL) and zero-shot cross-lingual transfer tasks with large-scale real datasets.




(a) MTL (b) GradNorm (c) PCGrad (d) GradVac

(e) RecAdam (f) Reptile (g) Seq.Reptile (h) Cos. sim. btw task gradients.

Figure 3: **(a)**∼**(g)** Loss surface and learning trajectory of each method. **(h)** Heatmap shows average pair-wise cosine similarity between the task gradients.

**Baselines** We compare our method against the following relevant baselines.

1. **STL:** Single Task Learning (STL) model trained on each single language.
2. **MTL:** Base MTL model of which objective is Eq. 1 without the regularizer $\Omega(\phi)$.
3. **GradNorm (Chen et al., 2018):** This model tackles task imbalance problem in MTL. It prevents the training from being dominated by a single task by adaptively weighting each task gradient.
4. **PCGrad (Yu et al., 2020):** This model aims to optimize MTL objective more aggressively by resolving gradient conflicts. Specifically, it projects a task gradient onto the other task gradients if the inner product between them is negative.
5. **GradVac (Wang et al., 2021):** Similarly to PCGrad, this model alters the task gradients to match the empirical moving average of cosine similarity between the task gradients.
6. **RecAdam (Chen et al., 2020):** A model trained with RecAdam optimizer to prevent catastrophic forgetting by penalizing $\ell_2$ distance from the the pretrained model.
7. **Reptile (Nichol et al., 2018):** A first-order meta-learning method suitable for large-scale learning scenario. Unlike our method, Reptile performs inner-optimization individually for each task.
8. **Sequential Reptile:** Our method that can align gradients across the tasks by composing the inner-learning trajectory with all the tasks.

### 4.1 Synthetic Experiments

We first verify our hypothesis with the following synthetic experiments. We define three local optima $x_1 = (0, 10), x_2 = (0, 0)$, and $x_3 = (10, 0)$ in a 2-dimensional space. Then, we define three tasks as minimizing each of the following loss functions with respect to $\phi \in \mathbb{R}^2$.

$$\mathcal{L}_i(\phi) = -200 \cdot \exp\left(0.2 \cdot \|\phi - x_i\|_2\right), \quad \text{for} \quad i = 1, 2, 3.$$

MTL objective is defined as $\sum_{i=1}^{3} \mathcal{L}_i(\phi)$, which we optimize from the initialization $(20, 5)$.

**Results and analysis** Fig. 3 shows the MTL loss surface and the learning trajectory of each method. We can see from the figure that except for Reptile and Sequential Reptile, all the other baselines converge to one of the MTL local minima, failing to find a reasonable solution that may generalize better across the tasks. While Reptile can avoid such a minimum, the resultant solution has very low cosine similarity (See Fig. 3h) because it does not enforce gradient alignments between tasks. On the other hand, Figure 3h shows that our Sequential Reptile tends to find a reasonable trade-off between minimizing MTL loss and maximizing cosine similarity, thanks to the implicit enforcement of inter-task gradient alignment in Eq. 7.

Table 1: F1 and EM score on the TYDI-QA dataset for QA. The best result for multilingual models is marked with bold while the underline denotes the best result among all the models including the monolingual model.

| | **Question Answering** (F1/EM) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **ar** | **bn** | **en** | **fi** | **id** | **ko** | **ru** | **sw** | **te** | **Avg.** |
| **STL** | 80.5 / 65.9 | 70.9 / 58.4 | 72.0 / 59.2 | 76.2 / 63.7 | 82.7 / 70.6 | 61.0 / 50.8 | 73.4 / 56.5 | 78.4 / 70.1 | 81.1 / 66.4 | 75.1 / 62.4 |
| **MTL** | 79.7 / 64.7 | 74.7 / 64.0 | 72.8 / 61.1 | 77.8 / 64.7 | 82.9 / 71.3 | 64.0 / 53.4 | 73.9 / 57.1 | 80.5 / 72.5 | 82.5 / 67.9 | 76.5 / 64.1 |
| **RecAdam** | 79.0 / 63.7 | 72.5 / 62.4 | 73.5 / 62.8 | 76.9 / 64.9 | 82.1 / 72.2 | 64.6 / 54.5 | 73.9 / 57.6 | 80.4 / 72.9 | 82.8 / 68.4 | 76.2 / 64.4 |
| **GradNorm** | 78.8 / 62.9 | 72.5 / 61.9 | 73.4 / 60.6 | 78.5 / 65.8 | 83.7 / 74.3 | 66.2 / 53.9 | 74.5 / 57.7 | 80.7 / 72.7 | **83.3** / **68.9** | 76.8 / 64.3 |
| **PCGrad** | 79.9 / 65.0 | 72.6 / 61.5 | 74.6 / 62.3 | 78.3 / 65.7 | 82.7 / 73.0 | 65.9 / 56.1 | 74.2 / 57.3 | 80.6 / 73.0 | 82.5 / 68.1 | 76.8 / 64.7 |
| **GradVac** | 80.1 / 64.7 | 71.5 / 59.2 | 73.0 / 61.2 | 78.6 / 65.5 | 83.1 / 72.4 | 63.8 / 53.4 | 74.1 / 57.6 | 80.6 / 72.6 | 82.2 / 67.5 | 76.3 / 63.8 |
| **Reptile** | 79.8 / 65.2 | **75.1** / **64.1** | 74.0 / 62.8 | 78.8 / 65.3 | 83.8 / 73.6 | 65.7 / 55.9 | 75.5 / 58.7 | 81.4 / 72.9 | 83.1 / 68.5 | 77.5 / 65.2 |
| **Seq.Reptile** | **81.2** / **66.7** | 73.9 / 62.6 | **76.7** / **65.2** | **79.4** / **66.3** | **84.9** / **74.7** | **68.0** / **58.2** | **76.8** / **59.2** | **82.9** / **74.8** | 83.0 / 68.6 | **78.5** / **66.3** |

Table 2: F1 score on WikiAnn dataset for NER. The best result for multilingual models is marked with bold while the underline denotes the best result among all the models including the monolingual model.

| | **Named Entity Recognition** (F1) | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **de** | **en** | **es** | **hi** | **jv** | **kk** | **mr** | **my** | **sw** | **te** | **tl** | **yo** | **Avg.** |
| **STL** | 90.3 | 85.0 | 92.0 | 89.7 | 59.1 | 88.5 | 89.4 | 61.7 | 90.7 | 80.1 | 96.3 | 77.7 | 83.3 |
| **MTL** | 83.4 | 77.8 | 87.6 | 82.3 | 77.7 | 87.5 | 82.2 | 75.7 | 87.5 | 78.8 | 83.5 | 90.8 | 82.9 |
| **RecAdam** | 84.5 | 80.0 | 88.5 | 82.7 | **85.3** | 88.5 | 84.4 | 70.3 | 89.0 | 81.6 | 87.7 | 91.6 | 84.5 |
| **GradNorm** | 83.6 | 77.5 | 87.3 | 82.8 | 78.3 | 87.8 | 81.3 | 73.5 | 85.4 | 78.9 | 83.6 | 91.4 | 82.6 |
| **PCGrad** | 83.8 | 78.5 | 88.1 | 81.7 | 79.7 | 87.8 | 81.7 | 74.4 | 85.9 | 78.4 | 85.7 | 92.3 | 83.1 |
| **GradVac** | 83.9 | 79.5 | 88.3 | 81.8 | 80.6 | 87.5 | 82.2 | 73.9 | 87.9 | 79.4 | 87.9 | **93.0** | 83.8 |
| **Reptile** | 85.9 | 82.4 | 90.0 | 86.3 | 81.3 | 81.4 | 86.8 | 61.8 | 90.6 | 72.7 | 92.8 | **93.0** | 83.7 |
| **Seq.Reptile** | **87.4** | **83.9** | **90.8** | **88.1** | 85.2 | **89.4** | **88.9** | **76.0** | **91.5** | **82.5** | **94.7** | 92.5 | **87.5** |

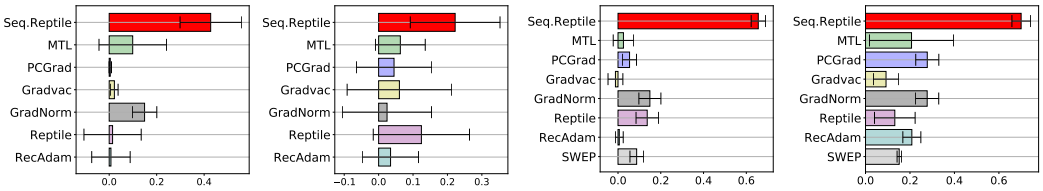
### 4.2 Multi-Task Learning

We next verify our method with large-scale real datasets. We consider multi-task learning tasks such as multilingual Question Answering (QA) and Named Entity Recognition (NER). For QA, we use "Gold passage" of TYDI-QA (Clark et al., 2020) dataset where a QA model predicts a start and end position of answer from a paragraph for a given question. For NER, we use WikiAnn dataset (Pan et al., 2017) where a model classifies each token of a sentence into three classes — Beginning of entity (B), Inside of entity (I), and Outside of entity (O). We consider each language as a distinct task and train MTL models.

**Implementation Details** For all the experiments, we use multilingual BERT (Devlin et al., 2019) base model as a backbone network. We fintune it with AdamW (Loshchilov & Hutter, 2019) optimizer, setting the inner-learning rate $\alpha$ to $3 \cdot 10^{-5}$. We use batch size 12 for QA and 16 for NER, respectively. For our method, we set the outer learning rate $\eta$ to 0.1 and the number inner-steps $K$ to 1000. Following Wang et al. (2021), for all the baselines, we sample eight tasks proportional to $p_t \propto (N_t)^{1/5}$ where $N_t$ is the number of training instances for task $t$. For our Sequential Reptile, we set the parameter of the categorical distribution in Eq. 5 to the same $p_t$.

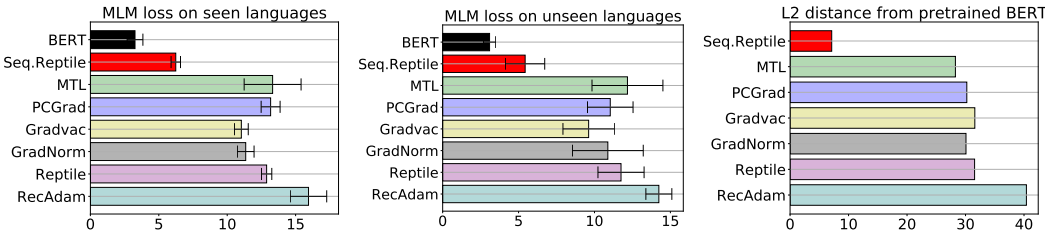
**Results** We compare our method, Sequential Reptile against the baselines on QA and NER tasks. Table 1 shows the results of QA task. We see that our method outperforms all the baselines including STL on most of the languages. Interestingly, all the other baselines but ours underperform STL on Arabic language which contains the largest number of training instances ($2 \sim 3$ times larger than the other languages. See Appendix B for more information about data statistics). It implies that the baselines suffer from negative transfer while ours is relatively less vulnerable. We can observe essentially the same tendency for NER task, which is a highly imbalanced such that the low-resource languages can have around 100 training instance, while the high-resource languages can have around 5,000 to 20,000 examples. Table 2 shows the results of NER task. We see that all the baselines but ours highly degrade the performance on high resource languages — de, en, es, hi, mr and tl, which means that they fail to address the imbalance problem properly and suffer from severe negative transfer. Even GradNorm is not effective in our experiments, which is developed to tackle the task imbalance problem by adaptively scaling task losses.

**Analysis** We next analyze the source of performance improvements. Our hypothesis is that our method can effectively filter out language specific knowledge when solving the downstream tasks, which prevents negative transfer and helps retain linguistic knowledge acquired from the pretraining.




(a) TYDI-QA (b) NER (c) SQuAD (d) XNLI

Figure 4: Average pair-wise cosine similarity between the gradients computed from different tasks.




(a) (b) (c)

Figure 5: **(a, b)** Average MLM loss on **(a)** seen and **(b)** unseen languages from Common Crawl dataset. We mask 15% tokens of sentences from Common Crawl dataset, which is preprocessed and provided by Wenzek et al. (2020), and compute the masked language modeling loss (MLM), which is reconstruction loss of the masked sentences. **(c)** $\ell_2$ distance between the finetuned models and the initially pretrained BERT.

Table 3: We evaluate the model, trained with TYDI-QA dataset, on five unseen languages from MLQA dataset.

| **TYDI-QA** $\rightarrow$ **MLQA** (F1/EM) | | | | | | |
|---|---|---|---|---|---|---|
| **Method** | **de** | **es** | **hi** | **vi** | **zh** | **Avg.** |
| **MTL** | 50.6 / 35.8 | 54.3 / 35.4 | 45.0 / 31.2 | 53.7 / 34.8 | 52.5 / 32.0 | 51.2 / 33.8 |
| **RecAdam** | 49.5 / 36.1 | 52.8 / 35.6 | 41.7 / 29.0 | 52.2 / 34.6 | 49.8 / 29.9 | 49.2 / 33.0 |
| **GradNorm** | 51.7 / 36.6 | 54.9 / 36.3 | 44.4 / 30.4 | 55.3 / 37.1 | 52.9 / 32.2 | 51.8 / 34.5 |
| **PCGrad** | 50.6 / 36.5 | 54.5 / 36.2 | 44.1 / 31.2 | 54.4 / 34.8 | 52.4 / 31.5 | 51.1 / 34.2 |
| **GradVac** | 50.0 / 35.4 | 53.0 / 35.2 | 41.6 / 28.6 | 52.9 / 34.9 | 51.3 / 31.2 | 49.8 / 33.1 |
| **Reptile** | 52.2 / 38.2 | 56.1 / 38.0 | 45.9 / 32.1 | 56.9 / 38.2 | 53.9 / 33.2 | 53.0 / 35.9 |
| **Seq.Reptile** | **53.7 / 38.5** | **57.6 / 38.9** | **47.7 / 33.7** | **58.1 / 39.2** | **55.1 / 34.7** | **54.4 / 37.0** |

Firstly, we quantitatively measure the cosine similarity between the gradients computed from different tasks in Fig. 4a (QA) and Fig. 4b (NER). We see from the figure that our Sequential Reptile shows the highest cosine similarity as expected, due to the approximate learning objective in Eq. 7. Such high cosine similarity between the task gradients implies that the model has captured the common knowledge well transferable across the languages. This is in contrast to the other baselines whose task gradients can have even negative cosine similarity with high probability. Such different gradient directions mean that the current model has memorized the knowledge not quite transferable across the languages, which can cause negative transfer from one language to others.

As a result of such negative transfer, we see from Fig. 5a and Fig. 5b that the baselines suffer from catastrophic forgetting. In those figures, high MLM losses on seen (Fig. 5a) and unseen languages (Fig. 5b) mean that the models have forgotten how to reconstruct the masked sentences, which is the original training objective of BERT. On the other hand, our method shows relatively lower MLM loss, demonstrating its effectiveness in preventing negative transfer and thereby alleviating the catastrophic forgetting. We further confirm this tendency in Fig. 5c by meausring the $\ell_2$ distance from the initially pretrained BERT model. We see that the distance is much shorter for our method than the baselines, which implies that the model trained with our method can better retain the linguistic knowledge than the baselines.

We can actually test if the finetuned models have acquired knowledge transferable across the languages by evaluating on some unseen languages without further finetuning. In Table 3, we test the finetuned QA models on five unseen languages from MLQA dataset (Lewis et al., 2020b). Again, Sequential Reptile outperforms all the baselines. The results confirm that the gradient alignment between tasks is key for obtaining common knowledge transferable across the languages.

Table 4: Zero-shot cross lingual transfer from English (SQuAD) to six unseen languages (MLQA).

| **SQuAD → MLQA** (F1/EM) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Method** | **ar** | **de** | **es** | **hi** | **vi** | **zh** | **Avg.** |
| **MTL** | 48.0 / 29.9 | 59.1 / 44.6 | 64.2 / 46.2 | 44.6 / 29.0 | 56.8 / 37.8 | 55.2 / 34.7 | 54.6 / 37.0 |
| **RecAdam** | 47.3 / 29.8 | 58.8 / 44.0 | 64.3 / 46.2 | 45.5 / 29.9 | 57.9 / 38.4 | 54.8 / 34.1 | 54.7 / 37.0 |
| **GradNorm** | 48.7 / 31.3 | 59.8 / 44.6 | 64.8 / 46.3 | 47.2 / 31.2 | 57.2 / 37.8 | 55.0 / 33.9 | 55.4 / 37.5 |
| **PCGrad** | 47.7 / 29.8 | 59.2 / 44.1 | 65.4 / 46.1 | 41.0 / 26.0 | 57.4 / 37.4 | 54.6 / 34.0 | 54.4 / 36.2 |
| **GradVac** | 45.3 / 28.3 | 58.2 / 43.4 | 63.9 / 45.9 | 42.4 / 27.9 | 56.3 / 37.1 | 53.0 / 32.9 | 53.1 / 35.9 |
| **SWEP** | 49.5 / 31.0 | 60.5 / **46.2** | 65.0 / 47.3 | 47.6 / 31.9 | 57.9 / 38.8 | 56.9 / 36.3 | 56.2 / 38.5 |
| **Reptile** | 46.8 / 29.9 | 58.7 / 44.4 | 65.1 / 47.5 | 41.5 / 27.9 | 56.1 / 37.7 | 53.9 / 33.9 | 53.6 / 36.8 |
| **Seq.Reptile** | **52.8 / 34.3** | **60.7** / 45.9 | **67.1 / 48.7** | **50.6 / 35.6** | **60.7 / 40.8** | **57.3 / 36.5** | **58.2 / 39.3** |

Table 5: Zero shot cross lingual transfer from English (MNLI) to fourteen unseen languages (XNLI).

| **MNLI → XNLI** (Accuracy) | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **ar** | **bg** | **de** | **el** | **es** | **fr** | **hi** | **ru** | **sw** | **th** | **tr** | **ur** | **vi** | **zh** | **Avg.** |
| **MTL** | 65.1 | 68.8 | 71.3 | 66.4 | 74.3 | 72.6 | 59.2 | 68.8 | 50.1 | 52.8 | 61.8 | 57.9 | 69.4 | 69.0 | 64.8 |
| **RecAdam** | 63.5 | 66.9 | 69.4 | 65.2 | 72.7 | 72.2 | 58.8 | 67.3 | 49.9 | 51.9 | 61.0 | 56.5 | 67.9 | 67.7 | 63.6 |
| **GradNorm** | 64.0 | 68.2 | 70.6 | 67.0 | 74.1 | 72.9 | 58.8 | 68.0 | 48.9 | 53.2 | 61.0 | 56.8 | 70.3 | 69.3 | 64.5 |
| **PCGrad** | 64.0 | 68.7 | 69.8 | 66.5 | 74.3 | 71.8 | 59.4 | 68.3 | 51.0 | 53.1 | 60.7 | 57.4 | 69.7 | 69.3 | 64.5 |
| **GradVac** | 62.3 | 67.8 | 69.2 | 65.9 | 72.6 | 72.2 | 59.6 | 67.1 | 51.1 | 52.9 | 61.7 | 56.4 | 68.8 | 68.0 | 63.9 |
| **SWEP** | 64.8 | **69.4** | 70.5 | 67.1 | 74.8 | **74.4** | 58.7 | **69.7** | 49.2 | 53.7 | 60.2 | 57.1 | 69.8 | 68.4 | 64.8 |
| **Reptile** | 63.3 | 66.6 | 69.4 | 64.9 | 72.6 | 71.3 | 58.3 | 66.9 | 46.4 | 47.5 | 58.6 | 55.9 | 68.6 | 66.9 | 62.6 |
| **Seq.Reptile** | **67.2** | 69.3 | **71.9** | **67.8** | **75.1** | 74.1 | **60.6** | 69.5 | **51.2** | **55.1** | **63.8** | **59.1** | **70.8** | **69.6** | **66.0** |

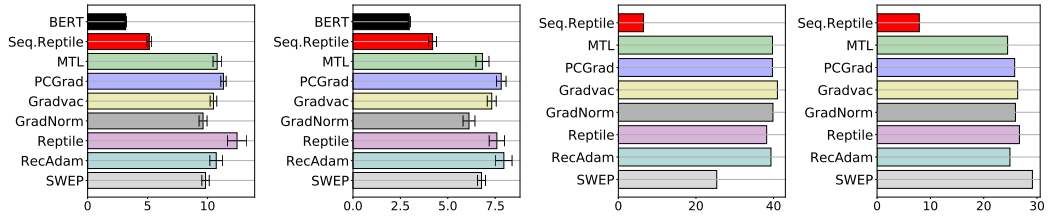
### 4.3 Zero-shot Cross Lingual Transfer

We next validate our method on zero-shot cross lingual transfer task, motivated from the previous results that our method can learn common knowledge well transferable across the languages. We train a model only with English annotated data and evaluate the model on target languages without further finetuning on the target datasets. In order to utilize the methods of MTL, we cluster the data into four groups with Gaussian mixture model. We focus on QA and natural language inference (NLI) task where, given a premise and hypothesis, a model predicts whether the premise entails the hypothesis, contradicts the hypothesis, or neither (neutral). For QA, we train the model on SQuAD (Rajpurkar et al., 2016) dataset and evaluate the model on six languages from MLQA dataset (Lewis et al., 2020b). For NLI, we use MNLI (Williams et al., 2018) dataset as a source training dataset and test the model on fourteen languages from XNLI (Conneau et al., 2018) as a target languages.

**Baselines** As well as the baselines from the previous experiments, we additionally include **SWEP** proposed by Lee et al. (2021). It learns to perturb word embeddings of pretrained language model and uses the perturbed input as additional training data, which is empirically shown to be robust against out-of distribution data.

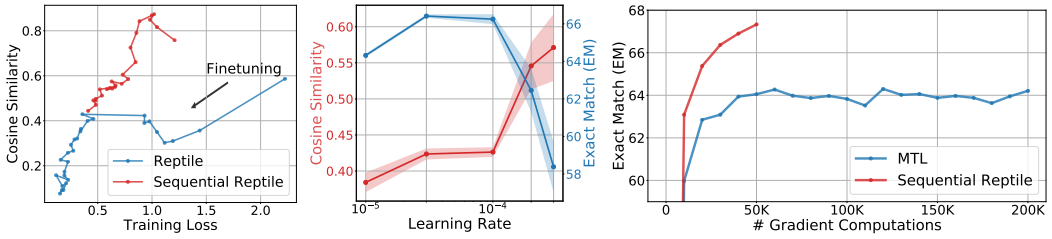
**Implementation Detail** We finetune multilingual BERT-base model with AdamW optimizer. For QA, we use the same hyperparameter in multi-task learning experiment. For NLI, we use batch size 32 and choose the learning rate $3 \cdot 10^{-5}$ or $5 \cdot 10^{-5}$ with AdamW optimizer based on the performance on the validation set. For our model, we set the outer learning rate $\eta$ to $0.1$ and the number inner steps $K$ to 1000. In order to construct multiple tasks from a single dataset, we cluster concatenation of questions and paragraphs from SQuAD or sentences from MNLI into four groups. Following Aharoni & Goldberg (2020), we encode the paragraphs or sentences into hidden representations with pretrained multilingual BERT. Then, we reduce the dimension of hidden representations with PCA and run Gaussian mixture model to partition them into disjoint four clusters. Since the number of training instances are almost evenly distributed for each task, we sample mini-batches from all the four tasks for all the baselines and set $p_t = 1/4$ for our model.

**Results and Analysis** The results of zero-shot cross lingual transfer show essentially the same tendency as those of multi-task learning in the previous subsection. Firstly, our model largely outperforms all the baselines for QA and NLI tasks as shown in Table 4 and 5. To see where the performance improvements come from, we compute the average pairwise cosine similarity between the gradients computed from different tasks in Fig. 4c and 4d. Again, Sequential Reptile shows much higher similarity than the baselines, which implies that our method can effectively filter out




(a) QA (b) NLI (c) QA (d) NLI

Figure 6: **(a,b)** Masked Language Modeling (MLM) loss. **(c,d)** $\ell_2$ distance from the pretrained BERT model.




(a) (b) (c)

Figure 7: **(a) Trade-off shown in Fig. 1:** average cosine similarity between task gradients vs. MTL training loss. **(b) Effect of the strength of gradient alignment:** Average cosine similarity between task gradients and test performance (EM) vs. inner-learning rate. **(c) Computational efficiency:** Test performance (EM) vs. the cumulative count of (inner-) gradient steps used for training.

task-specific knowledge incompatible across the languages and thereby prevent negative transfer. As a result, Fig. 6a and 6b show that our method can better retain the linguistic knowledge obtained from the pretraining in terms of relatively lower MLM loss. Further, Fig. 6c and 6d confirm this tendency with shorter $\ell_2$ distance from the initially pretrained BERT model.

### 4.4 Further Analysis

**(a) Trade-off shown in Fig. 1:** In Fig. 7a, MTL loss and cosine similarity decrease as we finetune the model from top-right to the bottom-left corner. In the meanwhile, our Sequential Reptile shows much higher cosine similarities between task gradients at the points of similar MTL losses, achieving better trade-off than Reptile. It explains why simple early stopping cannot outperform Sequential Reptile (see Fig. 7c) that directly enforces gradient alignment across tasks.

**(b) Effect of the strength of gradient alignment:** Then the next question is, how can we further control the trade-off to maximize performance? According to Eq. 7, we can strengthen or weaken the gradient alignment by increasing or decreasing the inner-learning rate $\alpha$, respectively. Fig. 7b shows that while we can control the cosine similarity by varying $\alpha$ as expected, the best-performing $\alpha$ is around $3 \cdot 10^{-5}$, which is indeed the most commonly used value for finetuning the BERT model.

**(c) Computational efficiency:** Lastly, one may suspect that training Sequential Reptile may take significantly longer wall clock time because task-specific optimizations are not parallelizable as in Reptile (See Fig. 2). However, this is not true. Fig. 7c shows that whereas base MTL requires around $40$K gradient computations to achieve 64 EM score, Sequential Reptile requires only around $15$K. As a result, although we run Sequential Reptile with a single GPU at a time, the wall-clock time becomes even comparable to the base MTL that we run in parallel with $8$ GPUs (only around $40\%$ increase).

## 5 Conclusion

In this paper, we showed that when finetuning a well-pretrained language model, it is important for the given set of downstream tasks to align gradients between them in order to prevent negative transfer and retain linguistic knowledge acquired from the pretraining. We then proposed a simple yet effective method that can align gradients between tasks in an efficient manner. Specifically, instead of performing multiple inner-optimizations separately for each task, we performed a single inner-optimization by sequentially sampling batches from all the tasks, followed by a Reptile outer update. We extensively validated the efficacy of our method on various MTL and zero-shot cross-lingual transfer tasks, where our method largely outperformed all the baselines we considered.

## REFERENCES

Roee Aharoni and Yoav Goldberg. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7747–7763, 2020.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*, 2019.

Steven Cao, Nikita Kitaev, and Dan Klein. Multilingual alignment of contextual word representations. In *International Conference on Learning Representations*, 2019.

Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7870–7881, 2020.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, pp. 794–803. PMLR, 2018.

Jonathan H Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470, 2020.

Alexis Conneau and Guillaume Lample. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems*, 32:7059–7069, 2019.

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

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, 2019.

Gunshi Gupta, Karmesh Yadav, and Liam Paull. Look-ahead meta learning for continual learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Junjie Hu, Melvin Johnson, Orhan Firat, Aditya Siddhant, and Graham Neubig. Explicit alignment objectives for multilingual bidirectional encoders. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3633–3643, 2021.

Zhuoliang Kang, Kristen Grauman, and Fei Sha. Learning with whom to share in multi-task feature learning. In *ICML*, 2011.

Abhishek Kumar and Hal Daumé III. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 1723–1730, 2012.

Giwoong Lee, Eunho Yang, and Sung Hwang. Asymmetric multi-task learning based on task relatedness and loss. In *International conference on machine learning*, pp. 230–238. PMLR, 2016.

Seanie Lee, Minki Kang, Juho Lee, and Sung Ju Hwang. Learning to perturb word embeddings for out-of-distribution QA. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pp. 5583–5595. Association for Computational Linguistics, 2021.

Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. Pre-training via paraphrasing. *Advances in Neural Information Processing Systems*, 33, 2020a.

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

Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. *Advances in neural information processing systems*, 32:12060–12070, 2019.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020.

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

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Farhad Nooralahzadeh, Giannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. Zero-shot cross-lingual transfer with meta learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4547–4562, 2020.

Lin Pan, Chung-Wei Hang, Haode Qi, Abhishek Shah, Saloni Potdar, and Mo Yu. Multilingual bert post-pretraining alignment. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 210–219, 2021.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1946–1958, 2017.

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

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, , and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 525–536, 2018.

Samuel L Smith, Benoit Dherin, David Barrett, and Soham De. On the origin of implicit regularization in stochastic gradient descent. In *International Conference on Learning Representations*, 2021.

Xinyi Wang, Yulia Tsvetkov, and Graham Neubig. Balancing training for multilingual neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8526–8537, 2020a.

Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime G. Carbonell. Characterizing and avoiding negative transfer. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 11293–11302. Computer Vision Foundation / IEEE, 2019a.

Zirui Wang, Jiateng Xie, Ruochen Xu, Yiming Yang, Graham Neubig, and Jaime G Carbonell. Cross-lingual alignment vs joint training: A comparative study and a simple unified framework. In *International Conference on Learning Representations*, 2019b.

Zirui Wang, Zachary C Lipton, and Yulia Tsvetkov. On negative interference in multilingual language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4438–4450, 2020b.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *International Conference on Learning Representations*, 2021.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Édouard Grave. Ccnet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 4003–4012, 2020.

Adina Williams, Nikita Nangia, and Samuel Bowman. 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)*, pp. 1112–1122, 2018.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 483–498, 2021.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Yu Zhang and Dit-Yan Yeung. A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pp. 733–742, 2010.

## A ALGORITHM

We provide the pseudocode for Sequential Reptile described in section 3.1:

**Algorithm 1** Sequential Reptile

**Input:** pretrained language model parameter $\phi$, a set of task-specific data $\{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$, probability vector $(p_1, \ldots, p_T)$ for categorical distribution, the number of inner steps $K$, inner step-size $\alpha$, outer-step size $\eta$.
**while** not converged **do**
  $\theta^{(0)} \leftarrow \phi$
  **for** $k = 1$ to $k = K$ **do**
    Sample a task $t_k \sim \text{Cat}(p_1, \ldots, p_T)$
    Sample a mini-batch $\mathcal{B}_{t_k}^{(k)}$ from the dataset $D_{t_k}$
    $\theta^{(k)} \leftarrow \theta^{(k-1)} - \alpha \frac{\partial \mathcal{L}(\theta^{(k-1)}; \mathcal{B}_{t_k}^{(k)})}{\partial \theta^{(k-1)}}$
  **end for**
  $\text{MG}(\phi) = \phi - \theta^{(K)}$
  $\phi \leftarrow \phi - \eta \cdot \text{MG}(\phi)$
**end while**

Table 6: The number of train/validation instances for each language from TYDI-QA dataset.

| Split | ar | bn | en | fi | id | ko | ru | sw | te | Total |
|---|---|---|---|---|---|---|---|---|---|---|
| Train | 14,805 | 2,390 | 3,696 | 6,855 | 5,702 | 1,625 | 6490 | 2,755 | 5,563 | 49,881 |
| Val. | 1,034 | 153 | 449 | 819 | 587 | 306 | 914 | 504 | 828 | 5,594 |

Table 7: The number of train/validation instances for each language from WikiAhn dataset.

| Split | de | en | es | hi | jv | kk | mr | my | sw | te | tl | yo | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train | 20,000 | 20,017 | 20,000 | 5,001 | 100 | 1,000 | 5,000 | 106 | 1,000 | 10,000 | 100 | 100 | 82,424 |
| Val. | 10,000 | 10,003 | 10,000 | 1,000 | 100 | 1,000 | 1,000 | 113 | 1,000 | 1,000 | 1,000 | 100 | 36,316 |

## B DATASET

**TYDI-QA (Clark et al., 2020)** It is multilingual question answering (QA) dataset covering 11 languages, where a model retrieves a passage that contains answer to the given question and find start and end position of the answer in the passage. Since we focus on extractive QA tasks, we use "Gold Passage" [1] in which ground truth paragraph containing answer to the question is provided. Since some of existing tools break due to the lack of white spaces for Thai and Japanese, the creators of the dataset does not provide the Gold Passage of those two languages. Following the conventional preprocessing for QA, we concatenate a question and paragraph and tokenize it with BertTokenizer (Devlin et al., 2019) which is implemented in transformers library (Wolf et al., 2020). We split the tokenized sentences into chunks with overlapping words. In Table 6, we provide the number of preprocessed training and validation instances for each language.

**WikiAhn (Pan et al., 2017)** It a multilingual named entity recognition dataset which is automatically constructed from Wikipedia. We provide the number of instances for training and validation in Table 7.

[1] https://github.com/google-research-datasets/tydiqa/blob/master/gold_passage_baseline/README.md

Table 8: The number of train/validation instances for each language from MLQA dataset.

| Split | ar | de | en | es | hi | vi | zh | Total |
|---|---|---|---|---|---|---|---|---|
| Val. | 517 | 512 | 1,148 | 500 | 507 | 511 | 504 | 4,199 |
| Test | 5,335 | 4,517 | 11,590 | 5,253 | 4,918 | 5,495 | 5,137 | 42,245 |

**XNLI (Conneau et al., 2018)** It is the dataset for multilingual natural language inference task, which consists of 14 languages other than English. Since it targets for zero-shot cross-lingual transfer, there is no training instances. It provides 7,500 human annotated instances for each validation and test set.

**MLQA (Lewis et al., 2020b)** It is the dataset for zero-shot multilingual question answering task. As XNLI dataset, it only provides only validation and test set for 6 languages other than English. In Table 8, we provide data statistics borrowed from the original paper (Lewis et al., 2020b)

**Common Crawl 100 (Conneau et al., 2020; Wenzek et al., 2020)** It is multilingual corpus consisting of more than 100 language that is used for pretraining XLM-R (Conneau et al., 2020) model. We download the preprocessed corpus [2] provided by Wenzek et al. (2020). We sample 5,000 instances for each language and evaluate masked language modeling loss.

## C Implicit Gradient alignment of sequential Reptile

In this section, we provide derivation of implicit gradient alignment of Sequential Reptile in the equation 7. Firstly, we define the following terms from Nichol et al. (2018).

$$\theta^{(0)} = \phi \qquad \text{(initial point)} \tag{8}$$

$$g_{t_k} = \frac{\partial \mathcal{L}(\theta^{(k-1)}; \mathcal{B}_{t_k}^{(k)})}{\partial \theta^{(k-1)}} \quad \text{(gradient obtained during SGD with mini-batch } \mathcal{B}_{t_k}^{(k)}) \tag{9}$$

$$\theta^{(k)} = \theta^{(k-1)} - \alpha g_{t_k} \qquad \text{(sequence of parameter vectors)} \tag{10}$$

$$\overline{g}_{t_k} = \frac{\partial \mathcal{L}(\theta^{(k-1)}; \mathcal{B}_{t_k}^{(k)})}{\partial \phi} \quad \text{(gradient at initial point with mini-batch with } \mathcal{B}_{t_k}^{(k)}) \tag{11}$$

$$\overline{H}_{t_k} = \frac{\partial^2 \mathcal{L}(\theta^{(k-1)}; \mathcal{B}_{t_k}^{(k)})}{\partial \phi^2} \quad \text{(Hessian at initial point)} \tag{12}$$

where $\alpha$ denotes a learning rate for inner optimization. $t_k \in \{1, \ldots, T\}$ is a task index and $\mathcal{B}_{t_k}^{(k)}$ is a mini-batch sampled from the task $t_k$.

**Corollary.** *After $K$ steps of inner-updates, expectation of $\sum_{k=1}^{K} g_{t_k}$ over the random sampling of tasks approximates the following:*

$$\mathbb{E}\left[\sum_{k=1}^{K} g_{t_k}\right] \approx \sum_{k=1}^{K} \overline{g}_{t_k} - \frac{\alpha}{2} \sum_{k=1}^{K} \sum_{j=1}^{k-1} \left\langle \overline{g}_{t_k}, \overline{g}_{t_j} \right\rangle \tag{13}$$

where $\langle \cdot, \cdot \rangle$ denotes a dot-product in $\mathbb{R}^d$.

*Proof.* We borrow the key idea from the theorem of Reptile (Nichol et al., 2018) that task specific inner-optimization of Reptile implicitly maximizes inner products between mini-batch gradients

[2] http://data.statmt.org/cc-100/

within a task. First, we approximate the gradient $g_{t_k}$ as follows.

$$
\begin{aligned}
g_{t_k} &= \frac{\partial \mathcal{L}(\phi; \mathcal{B}_{t_k}^{(k)})}{\partial \phi} + \frac{\partial^2 \mathcal{L}(\phi; \mathcal{B}_{t_k}^{(k)})}{\partial \phi^2}(\theta^{(k)} - \phi) + O(\|\theta^{(k)} - \phi\|_2^2) \\
&= \overline{g}_{t_k} - \alpha \overline{H}_{t_k} \sum_{j=1}^{k-1} g_{t_j} + O(\alpha^2) \\
&= \overline{g}_{t_k} - \alpha \overline{H}_{t_k} \sum_{j=1}^{k-1} \overline{g}_{t_j} + O(\alpha^2)
\end{aligned}
$$

After $K$ gradient steps, we approximate the expectation of the meta-gradient $\text{MG}(\phi)$ over the task $t_1, \ldots, t_K$, where $t_k \sim \text{Cat}(p_1, \ldots, p_T)$ for $k = 1, \ldots, K$.

$$
\begin{aligned}
\mathbb{E}\left[\text{MG}(\phi)\right] &= \mathbb{E}\left[\sum_{k=1}^{K} g_{t_k}\right] \\
&\approx \mathbb{E}\left[\sum_{k=1}^{K} \overline{g}_{t_k} - \alpha \sum_{k=1}^{K}\sum_{j=1}^{k-1} \overline{H}_{t_k} \overline{g}_{t_j}\right] \\
&= \mathbb{E}\left[\sum_{k=1}^{K} \overline{g}_{t_k}\right] - \mathbb{E}\left[\frac{\alpha}{2}\left(\sum_{k=1}^{K}\sum_{j=1}^{k-1} \overline{H}_{t_k} \overline{g}_{t_j} + \sum_{k=1}^{K}\sum_{j=1}^{k-1} \overline{H}_{t_j} \overline{g}_{t_k}\right)\right] \\
&= \mathbb{E}\left[\sum_{k=1}^{K} \overline{g}_{t_k}\right] - \mathbb{E}\left[\frac{\alpha}{2}\sum_{k=1}^{K}\sum_{j=1}^{k-1} \frac{\partial \left\langle g_{t_k},\, g_{t_j}\right\rangle}{\partial \phi}\right] \\
&= \mathbb{E}\left[\sum_{k=1}^{K} \overline{g}_{t_k} - \frac{\alpha}{2}\sum_{k=1}^{K}\sum_{j=1}^{k-1} \frac{\partial \left\langle g_{t_k},\, g_{t_j}\right\rangle}{\partial \phi}\right] \\
&= \mathbb{E}\left[\sum_{k=1}^{K} \overline{g}_{t_k} - \frac{\alpha}{2}\frac{\partial}{\partial \phi}\left(\sum_{k=1}^{K}\sum_{j=1}^{k-1} \left\langle g_{t_k},\, g_{t_j}\right\rangle\right)\right] \\
&= \mathbb{E}\left[\frac{\partial}{\partial \phi}\left(\sum_{k=1}^{K} \mathcal{L}(\phi; \mathcal{B}_{t_k}^{(k)}) - \frac{\alpha}{2}\sum_{k=1}^{K}\sum_{j=1}^{k-1} \left\langle \frac{\partial \mathcal{L}(\phi; \mathcal{B}_{t_k}^{(k)})}{\partial \phi}, \frac{\partial \mathcal{L}(\phi; \mathcal{B}_{t_j}^{(j)})}{\partial \phi}\right\rangle\right)\right] \\
&= \frac{\partial}{\partial \phi}\mathbb{E}\left[\sum_{k=1}^{K} \mathcal{L}(\phi; \mathcal{B}_{t_k}^{(k)}) - \frac{\alpha}{2}\sum_{k=1}^{K}\sum_{j=1}^{k-1} \left\langle \frac{\partial \mathcal{L}(\phi; \mathcal{B}_{t_k}^{(k)})}{\partial \phi}, \frac{\partial \mathcal{L}(\phi; \mathcal{B}_{t_j}^{(j)})}{\partial \phi}\right\rangle\right]
\end{aligned}
$$

□

Similarly, Smith et al. (2021) show implicit regularization of stochastic gradient descent (SGD). After one epoch of SGD, it implicitly bias a model to minimize difference between gradient of each example and full batch gradient. As a result, it approximately aligns gradient of each instance with the full-batch gradient.