Less-Forgetting Multi-Lingual Fine-Tuning

Anonymous Author(s)

Abstract

Multi-lingual Language Models (e.g., mBERT) have the cross-lingual zero-shot transfer ability, i.e., after fine-tuning a multi-lingual model on a source language, the model can achieve promising zero-shot performance on a target language. This ability can help to tackle the challenges associated with low-resourced languages and has gained much attention recently. Existing works typically focus on the mono-lingual fine-tuning scenario, where only one source language involves in fine-tuning. However, in practice, there might be multiple high-resource languages can be used as source languages. It is natural to ask *whether fine-tuning a multi-lingual model on multiple source languages can improve the zero-shot ability?* and *how to do this kind of multi-lingual fine-tuning effectively?* Unfortunately, these issues remain unexplored. Accordingly, to fill this gap, this paper conducts both theoretical and experimental analysis on this multi-lingual fine-tuning setting; furthermore, we propose a multi-lingual fine-tuning method, Less-forgetting Multi-lingual Fine-tuning (LF-MLF), which can achieve superior zero-shot performance. The results of extensive experiments back up our theoretical analysis and validate the superiority of our proposals.

1 Introduction

Multi-lingual Language Models, which are pre-trained using unlabeled data from multiple languages (including both high-resource and low-resource languages), support zero-shot cross-lingual transfer learning; in other words, for a down-stream task, after fine-tuning a multi-lingual language model using the labelled task-specific training data of a source language, the model can achieve promising performance on the task-specific test data of a target language. It can reduce the need of task-specific labelled data for low-resource languages and give hope to low-resource language process. Thus, multi-lingual model-based zero-shot cross-lingual transfer learning has drawn large attention.

Existing works focus on the mono-lingual fine-tuning setting, where they fine-tune a multilingual model on one source language (typically English) to obtain a model that has proper zero-shot performance on target languages [1, 2]. By contrast, in practical down-stream tasks, it is common that there are multiple high-resource languages have labelled data available for fine-tuning. For example, in Name Entity Recognition (NER), it is easy to obtain labeled data for English, German, French and so on. It is natural to ask *whether fine-tuning a multi-lingual model on multiple source languages can improve the zero-shot ability?* However, this issue remains unexplored.

In this paper, our experimental analysis answers *yes* for the above question. Nevertheless, another question raises– *how to do this kind of multi-lingual fine-tuning effectively?* Because, multi-lingual fine-tuning, which has not been explored before, is quite different from the mono-lingual fine-tuning. Specifically, when fine-tuning on one language, we just need to update the model toward the gradient descent direction. By contrast, in the multi-lingual fine-tuning setting, we have to choose an update direction in a feasible set, which is the convex hull of the gradient descent directions of the included

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

languages. The problem becomes *which direction should be chosen?* This paper proposes to choose the *less-forgetting direction* to avoid the multi-lingual model forgetting the target languages, namely prevent the model from fitting the source languages too much to lose the generalization ability on the target languages; moreover, the direction should be a descent direction common to all the source languages. To ensure the less-forgetting and common-descent requirements, we formulate multi-lingual fine-tuning as a constrained optimization problem and propose an algorithm to solve this problem. The proposed algorithm is dubbed Less-forgetting Multi-lingual Fine-tuning (LF-MLF).

For LF-MLF, we have conducted both theoretical and experimental analysis. Theoretical results demonstrate that LF-MLF can effectively reduce the forgetting of multi-lingual models in fine-tuning; besides, LF-MLF can achieve the Pareto stationarity. The superiority of LF-MLF has been experimentally verified on three kind of tasks (i.e., Named Entity Recognition, Question Answering and Natural Language Inference). Furthermore, we also conduct experimental analysis on several factors that impact the performance of multi-lingual fine-tuning (e.g., the impact of the number of the fine-tuning languages), which have the potential to forge new trends in multi-lingual fine-tuning.

2 Related Works

Multi-lingual Language Models (MLLMs) are pre-trained using tons of unlabeled data from multiple languages and expected to own the cross-lingual zero-shot transfer ability. This ability can facilitate the low resource languages processing with knowledge transferred from the high resource languages [2]. Several MLLMs have been proposed, such as mBERT [3] and XLM-R [4]. Given a MLLM, existing works commonly analyze its cross-lingual zero-shot transfer ability by the means of fine-tuning the model on a source language and observing the model's zero-shot performance on the target languages [5, 6, 7]. The fine-tuning scenario where multiple source languages are involved in fine-tuning, namely multi-lingual fine-tuning, remains unexplored.

Continual Learning (CL) studies the problem of learning a model on sequential tasks without forgetting knowledge obtained from the preceding tasks [8]. In CL, reducing forgetting is the major challenge. A large number of methods have been proposed to reduce forgetting, such as GEM [9], EWC [10] and IMM [11]; however, these methods typically focus on the single-objective optimization setting, where there is only one learning objective in a task. None of them considers the multi-objective optimization setting, where there are multiple learning objective in a task. Nevertheless, multi-lingual fine-tuning involves a multi-objective optimization problem, which aims to optimize the fine-tuning performance on all the including languages. Thus, existing CL methods are not suitable for multi-lingual fine-tuning.

Multi-task Learning (MTL) simultaneously learns multiple tasks and aims to achieve proper performance on all the including tasks. It can be formulated as a multi-objective optimization problem [12]. Various methods have been proposed to solve this kind of multi-objective optimization problem [13, 14, 15, 16]; however, none of them is designed for the pre-training/fine-tuning paradigm, as well as they do not care about the zero-shot performance of a model. Therefore, MTL methods are not suitable for multi-lingual fine-tuning, where the zero-shot performance is indispensable.

3 Less-forgetting Multi-lingual Fine-tuning

Multi-lingual fine-tuning, where a multi-lingual language model is fine-tuned on task-specific data from a set of languages (defined as source languages), has two goals: (1) keep the cross-lingual zero-shot transfer ability of the multi-lingual language model and (2) achieving proper performance on the source languages. For the first goal, it is necessary to avoid the model from forgetting the cross-lingual knowledge it obtained in the pre-training phase. For the second goal, we face a multi-objective optimization problem, where the losses for the source languages should be jointly optimized. In this section, we firstly give a formal definition of the multi-lingual fine-tuning and then propose a method (dubbed LF-MLF) that can simultaneously achieve the above two goals.

3.1 Problem Definition of Multi-lingual Fine-tuning

Given a multi-lingual language model θ_p, which is pre-trained using large amounts of unlabeled data from language set S_p, we aim to fine-tune this model on a downstream task and achieve proper performance. For the downstream task, we have task-specific labeled training data from a set of source languages S_f, where $S_f \subseteq S_p$. After fine-tuning, θ_p has been updated as θ_f. Assume there are T languages in source languages set S_f. For each language, the task-specific loss is denoted as $L_t(\theta_f)$. Under the gradient descent paradigm of deep learning, in the kth iteration of multi-lingual fine-tuning, the model is updated by

$$\theta_k = \theta_{k-1} - \eta \sum_{t=1}^{T} w_t^k \nabla L_t(\theta_{k-1}). \tag{1}$$

where η is the learning rate, $\sum_{t=1}^{T} w_t^k = 1$ and $w_t^k \geq 0, t \in \{1, ..., T\}$. In Eq. (1), $-\sum_{t=1}^{T} w_t^k \nabla L_t(\theta_{k-1})$ is the convex hull of the gradient descent directions of the included languages, which contains feasible gradient descent directions for the multi-lingual fine-tuning.

Furthermore, denote the pre-training loss of θ_p and θ_f as $L_p(\theta_p)$ and $L_p(\theta_f)$ respectively. According to the classic definition of forgetting [17], we define the forgetting between θ_p and θ_f as $F_p = L_p(\theta_f) - L_p(\theta_p)$, which quantifies the performance degeneration of θ_f on the multi-lingual pre-training task (e.g., masking tokens). With the above notation, the two goals for multi-lingual fine-tuning can be formally defined as: (1) minimizing F_p and (2) minimizing $\{L_1(\theta_f), ..., L_T(\theta_f)\}$.

3.2 Upper Bound the Forgetting in Multi-lingual Fine-tuning

In fine-tuning, we cannot directly compute the forgetting F_p, because the pre-training the head (e.g., masked language modeling head) has already been replaced by downstream-task-specific head (e.g., a token classification head) and $L_p(\theta_f)$ is not accessible. Alternatively, we propose to minimize the forgetting by minimizing its upper bound. In Theorem 1, we give a upper bound for the forgetting of multi-lingual fine-tuning. The assumptions in Theorem 1 are widely accepted [17, 18, 19] and can ensure proper performance.

Theorem 1. *Assume $L_p(\theta_f)$ can be approximated by its second order Taylor expansion and θ_p is a minima w.r.t the pre-training loss. Then, we have*

$$F_p \leq \frac{\lambda_p \eta}{2} \| \sum_{k=1}^{K} \sum_{t=1}^{T} w_t^k \nabla L_t(\theta_{k-1}) \|^2. \tag{2}$$

where λ_p is the maximum eigenvalue of $\nabla^2 L_p(\theta_p)$, and K is the number of iterations of fine-tuning.

Proof. The proof is provided in the supplementary materials. □

3.3 Less-forgetting Multi-lingual Fine-tuning

In multi-lingual fine-tuning, we not only need to keep the multi-lingual memory of the pre-trained model but also need to achieve proper fine-tuning performance, i.e., minimizing F_p and $\{L_1(\theta_f), ..., L_T(\theta_f)\}$ together. Achieving these two goals at same time is challenging and remains unexplored. In this section, we propose a novel method, dubbed Less-forgetting Multi-lingual Fine-tuning (LF-MLF), to jointly achieve these two goals.

Specifically, we minimizing F_p by tightening its upper bound introduced in Theorem 1. Besides, minimizing $\{L_1(\theta_f), ..., L_T(\theta_f)\}$ is a multi-objective optimization (MOO) problem. We solve this MOO problem by the means of multi-objective gradient descent.

According to Theorem 1, in the kth iteration,

$$F_p^k \leq \frac{\lambda_p \eta}{2} \| \sum_{i=1}^{k} \sum_{t=1}^{T} w_t^i \nabla L_t(\theta_{i-1}) \|^2 = \frac{\lambda_p \eta}{2} \| \bar{\nabla} L + \sum_{t=1}^{T} w_t^k \nabla L_t(\theta_{k-1}) \|^2 \tag{3}$$

Algorithm 1: Less-forgetting Multi-lingual Fine-tuning (LF-MLF)

Input: Pre-trained model θ_p, Number of iterations K.
for $k \leftarrow 1$ **to** K **do**
  if k == 1 **then**
    Solve Problem. (2) with $c = [0, 0, ..., 0]^\top$ and obtain solution w_1^*.
    $\theta_1 = \theta_p - \eta \sum_{t=1}^{T} (w_1^*)^t \nabla L_t(\theta_p)$.
  else
    Solve Problem. (2) and obtain solution w_k^*.
    $\theta_k = \theta_{k-1} - \eta \sum_{t=1}^{T} (w_k^*)^t \nabla L_t(\theta_{k-1})$
  end if
  if the feasible set of Problem. (2) is empty **then**
    break.
  end if
end for
return θ_K.

where $\bar{\nabla L} = \sum_{i=1}^{k-1} \sum_{t=1}^{T} w_t^i \nabla L_t(\theta_{i-1})$ and $\theta_0 = \theta_p$.

To reduce fogetting, in the kth iteration, we propose to minimize F_p^k's upper bound as in Eq. (4).

$$\min_{w^k} \|\bar{\nabla L} + \sum_{t=1}^{T} w_t^k \nabla L_t(\theta_{k-1})\|^2 \tag{4}$$

Except for reducing forgetting, we also need to decrease the task-specific losses of the fine-tuned languages, i.e, move toward the objective of minimizing $\{L_1(\theta_f), ..., L_T(\theta_f)\}$. Thus, in the kth iteration, we need to update the model in a descent direction common to all the source languages, which is the principle of multi-objective gradient descent. Under this principle, the update direction should not have a obtuse angle with the gradient direction on each language as in Eq. (5).

$$(\sum_{t=1}^{T} w_t^k \nabla L_t(\theta_{k-1}))^\top \nabla L_t(\theta_{k-1}) \geq 0, \ for \ t \in \ \{1, ..., T\}. \tag{5}$$

To reduce fogetting and minimizing the downstream task losses, we combine Eq. (4) with Eq. (5) and propose to formulate the less-forgetting multi-lingual fine-tuning as the following constrained optimization problem.

Problem 1.

$$\begin{array}{ll} \min\limits_{w^k} & \|\bar{\nabla L} + \sum_{t=1}^{T} w_t^k \nabla L_t(\theta_{k-1})\|^2 \\ s.t. & (\sum_{t=1}^{T} w_t^k \nabla L_t(\theta_{k-1}))^\top L_t(\theta_{k-1}) \geq 0, t \in \{1, ..., T\} \\ & \sum_{t=1}^{T} w_t^k = 1 \\ & w_t^k \geq 0, t \in \{1, ..., T\}. \end{array} \tag{6}$$

Problem 1 can be formulated as the following quadratic programming problem with T variables and $2T + 1$ constraints.

Problem 2.

$$\begin{array}{ll} \min\limits_{w^k} & \frac{1}{2}(w^k)^\top G w^k + (w^k)^\top c \\ s.t. & b_t^\top w^k \geq 0, t \in \{1, ..., T\} \\ & \sum_{t=1}^{T} w_t^k = 1 \\ & w_t^k \geq 0, t \in \{1, ..., T\}. \end{array} \tag{7}$$

where $G = \nabla L(\theta_{k-1})^T \nabla L(\theta_{k-1})$, $\nabla L(\theta_{k-1}) = [\nabla L_1(\theta_{k-1}), L_2(\theta_{k-1}), ..., L_T(\theta_{k-1})]^\top$. $c = \nabla L(\theta_{k-1})^T \bar{\nabla L}$ and $b_t = \nabla L(\theta_{k-1})^\top \nabla L_t(\theta_{k-1})$.

Problem 2 can be effectively solved. This paper adopts a interior-point solver [20]. Let w_k^* be the solution of the above quadratic programming problem. In our proposed LF-MLF, the update rule is

$$\theta_k = \theta_{k-1} - \eta \sum_{t=1}^{T} (w_k^*)^t \nabla L_t(\theta_{k-1}). \tag{8}$$

Overall, the challenging multi-lingual fine-tuning problem is transferred to a simple quadratic programming problem. The detailed steps of LF-MLF can be referred to Algorithm 1.

4 Theoretical Analysis

In less-forgetting multi-lingual fine-tuning, we explicitly set the goal, minimizing the upper bound, as the optimization objective. Therefore, the less-forgetting property of the proposed LF-MLF method is straightforward. By contrast, the target of minimizing $\{L_1(\theta_f), ..., L_T(\theta_f)\}$ is achieved by the means of putting constraints on the update direction, which forces the update direction to be the common decreasing direction for all the fine-tuned languages. Its effectiveness seems not obvious. To make it clear, in this section, we conduct theoretical analysis on LF-MLF's fine-tuning performance.

In multi-lingual fine-tuning, we aim to jointly fine-tune several languages, namely minimizing the loss vector $\{L_1(\theta_f), ..., L_T(\theta_f)\}$. It is a multi-objective optimization problem, where the optimal solutions are Pareto optimal as in Definition 1.

Definition 1 (Pareto Optimality)**.** *For $\boldsymbol{L}(\theta) = [\{L_1(\theta), ..., L_T(\theta)\}]^\top$, the Pareto optimality w.r.t the optimization objective $\min_\theta \boldsymbol{L}(\theta)$ is defined as follows: (1) A solution θ dominates a solution $\overline{\theta}$ if $L_t(\theta) \leq L_t(\overline{\theta})$ for all $t \in T$ and $\boldsymbol{L}(\theta) \neq \boldsymbol{L}(\overline{\theta})$ and (2) A solution θ^* is deemed Pareto optimal if there exists no solution θ that dominates θ^*.*

However, in deep learning, the parameter spaces are highly non-convex. To the best of our knowledge, there is no method that can learn a Pareto optimal deep neural network model. Settling for less, the best we can do is to achieve the Pareto stationarity as in Definition 2. Pareto stationarity is a necessary condition of achieving Pareto optimality.

Definition 2 (Pareto Stationarity)**.** *For $\boldsymbol{L}(\theta^0) = [\{L_1(\theta^0), ..., L_T(\theta^0)\}]^\top$ is said to be Pareto-stationary at the design-point θ^0 iff there exists a convex combination of the gradient-vectors, $\sum_{t=1}^{T} w_t \nabla L_t(\theta^0)$, that is equal to zero:*

$$\sum_{t=1}^{T} w_t \nabla L_t(\theta^0) = 0; \quad \sum_{t=1}^{T} w_t^k = 1; \quad w_t^k \geq 0, \ t \in \{1, ..., T\}. \tag{9}$$

In Theorem 2 and 3, we propose that LF-MLF can achieve Pareto stationarity, which theoretically verifies that our proposed method can achieve proper performance on the fine-tuning languages.

Theorem 2. *Let $\mathcal{H}$ be a Hilbert space of finite dimension N. Let $L_t(\theta_k)$ $(1 \leq t \leq T \leq N)$ be T smooth functions of the vector $\theta_k \in \mathcal{H}$, and θ_k^0 a particular admissible design-point. Let w_k^* be the solution of Problem 2 and descent direction $\nabla L = \sum_{t=1}^{T} (w_k^*)^t \nabla L_t(\theta_k)$. Then, either $\nabla L = \varnothing$, and $[L_1(\theta_k^0), ..., L_T(\theta_k^0)]^\top$ are pareto stationary at θ_k^0 or $\nabla L \neq \varnothing$ and $-\nabla L$ is a descent direction common to all $\{L_t(\theta_k)\}_{t=1}^{T}$.*

Proof. The proof is provided in the supplementary materials. □

Theorem 3. *LF-MLF can stop after a finite number of iterations if a Pareto stationary point is reached. Otherwise, If the sequence of iterates $\{\theta_k\}_{k=1}^{K}$ of the LF-MLF is infinite, it admits a weakly convergent subsequence.*

Proof. The proof is provided in the supplementary materials. □

5 Experiments

In this section, we perform experimental studies on three downstream tasks– Named Entity Recognition (NER), Question Answering (QA) and Natural Language Inference (NLI) respectively to evaluate the performance of our proposed LF-MLF and verify our theoretical analysis. The implementation is based on PyTorch [21] and XTREME benchmark [22]. The code is attached in the supplementary materials.

5.1 Experimental Setup

Datasets: In our experiments, we adopt the NER [23], TyDiQA-GoldP [24] and XNLI [25] datasets for NER, QA and NLI respectively from the XTREME benchmark [22]. The details of these datasets are introuced in the supplementary material.

Baselines: We compare our proposed LF-MLF with the following baselines: (1) **Single-lingual Fine-tuning (SLF):** fine-tuning on each language independently; (2) **Uniform-MLF:** fine-tuning on the included languages simultaneously by means of a uniformly weighted sum of the language-specific gradient descent directions, i.e., $w_t^k = \frac{1}{T}$ constantly; (3) **PCGrad-MLF:** using the Project Conflicting Gradients (PCGrad) method proposed by [26] on multi-lingual fine-tuning; (4) **GradVac-MLF:** using the Gradient Vaccine (GradVac) method proposed by [27] on multi-lingual fine-tuning.

Experimental Settings: In our experiments, we generally fine-tune XLM-RoBERTa models (base-sized model) [28] with a training batch size of 32, and AdamW [29] is used as the optimizer. As to the numbers of fine-tuning epochs, we adopt the default setting of the XTREME benchmark, that are 10, 3, 5 for NER, TyDiQA and XNLI respectively. Besides, the learning rates are selected from $\{1e-5, 2e-5, 5e-5\}$, $\{1e-5, 2e-5, 5e-5\}$ and $\{5e-6, 1e-5, 2e-5\}$ with grid search for NER, TyDiQA and XNLI respectively. All the results are averaged over 3 runs.

5.2 Performance of Multi-lingual Fine-tuning

Conducting experiments on NER, TyDiQA and XNLI, we compare LF-MLF with the baselines on the zero-shot performance and fine-tuning performance respectively. For each dataset, we choose two or three source languages for fine-tuning according the following rules: (1) belongs to the Top 10 high-resource languages w.r.t Wikipedias; (2) the higher-resource the better. For NER and XNLI, *en, de, fr* are chosen, while *en, ru* are chosen for TyDiQA.

5.2.1 Zero-shot Performance

To evaluate the zero-shot performance of LF-MLF and the baselines, we record the average F1 score over the target languages that have not been fine-tuned (i.e,. the other 45 languages except *en, de, fr* in NER, the other 7 languages except *en, ru* in TyDiQA and the other 10 languages except *en, de, fr* in XNLI). The results are reported in Table 1, 2 and 3 for NER, TyDiQA and XNLI respectively. These results demonstrate: (1) comparing with the single lingual fine-tuning, multi-lingual language fine-tuning can significantly improve the zero-shot performance; (2) LF-MLF outperforms the baseline methods on all the three applications.

Method	en-SLF	de-SLF	fr-SLF	Uniform-MLF	PCGrad-MLF	GradVac-MLF	LF-MLF
Zero-Shot	60.05	58.53	63.97	67.45	67.40	67.57	**68.50**

Table 1: Zero-shot performance (average F1 score over the other 45 languages) on NER.

Method	en-SLF	ru-SLF	Uniform-MLF	PCGrad-MLF	GradVac-MLF	LF-MLF
Zero-Shot	55.05	60.66	62.51	61.74	61.36	**63.03**

Table 2: Zero-shot performance (average F1 score over the other 7 languages) on TyDiQA-GoldP.

Method	en-SLF	de-SLF	fr-SLF	Uniform-MLF	PCGrad-MLF	GradVac-MLF	LF-MLF
Zero-Shot	72.46	74.26	74.01	75.17	74.87	74.67	**75.56**

Table 3: Zero-shot performance (average F1 score over the other 12 languages) on XNLI.

5.2.2 Fine-tuning Performance

Except the zero-shot performance, we also care about the fine-tuning performance, namely the F1-score on the languages that have been fine-tuned. The results of fine-tuning performance are reported in Table 4, 5 and 6 for NER, TyDiQA and XNLI respectively. From these results, we can see that:

Method	en-SLF	de-SLF	fr-SLF	Uniform-MLF	PCGrad-MLF	GradVac-MLF	LF-MLF
en	82.59	–	–	**83.30**	81.76	82.07	82.75
de	–	88.78	–	87.90	87.13	87.30	**88.92**
fr	–	–	**89.75**	89.68	88.85	89.70	89.15

Table 4: Fine-tuning performance (average F1 score over the other 45 languages) on NER.

Method	en-SLF	ru-SLF	Uniform-MLF	PCGrad-MLF	GradVac-MLF	LF-MLF
en	66.03	–	68.71	69.28	67.26	**69.31**
ru	–	70.89	**72.43**	71.76	71.87	71.66

Table 5: Fine-tuning performance (average F1 score over the other 7 languages) on TyDiQA-GoldP.

Method	en-SLF	de-SLF	fr-SLF	Uniform-MLF	PCGrad-MLF	GradVac-MLF	LF-MLF
en	84.29	–	–	**84.93**	84.75	84.59	84.81
de	–	78.98	–	79.92	79.48	79.96	**80.01**
fr	–	–	79.80	80.63	79.98	80.43	**80.83**

Table 6: Fine-tuning performance (average F1 score over the other 45 languages) on XNLI.

(1) the multi-lingual methods do not clearly outperform single-lingual methods on corresponding languages; (2) LF-MLF's performance is comparable to the baselines, and there is no baseline method can dominate LF-MLF. PCGrad-MLF and GradVac-MLF have inferior performance comparing with Uniform-MLF and LF-MLF. It because that PCGrad-MLF and GradVac-MLF cannot guarantee to find a Pareto stationarity, while Uniform-MLF and LF-MLF can find a Pareto stationarity. It verifies the theoretical analysis proposed in Section 4.

Overall, from section 5.2.1 and 5.2.2, we can see that LF-MLF has superior zero-shot performance and comparable fine-tuning performance. It verifies that LF-MLF has superior cross-lingual transfer ability. The superiority comes from LF-MLF's less-forgetting property, because such property enables LF-MLF to remember more about the target languages.

5.3 Impact of the Number of the Source Languages

In the last section, it is concluded that multi-lingual fine-tuning can significantly improve the zero-shot performance. Then, it is natural to ask *whether the zero-shot performance can be further improved when more languages have been fine-tuned*? In this section, we experimentally present that *the answer is yes*. Besides, we also investigate the impact of the number of source languages on the fine-tuning performance. Our experiments are conducted on the NER dataset, and the source languages are selected from the Top 10 high-resource languages w.r.t Wikipedias, i.e., *en, de, fr, ru, es, ja, it, zh, ko, ar*. By contrast, the other thirty eight languages are used as target languages. In the experiments, we increase the source languages from Top 2 to Top 10 and record the changing of zero-shot and fine-tuning performance.

5.3.1 Impact on the Zero-shot Performance

Figure 1 illustrates the change of MLF's zero-shot performance along with the increasing number of source languages. From this figure, we can conclude that (1) generally speaking, the zero-shot performance of a multi-lingual model continuously improves with the increasing of the number of source languages; (2) LF-MLF consistently outperforms Uniform-MLF when the number of source languages changes.

5.3.2 Impact on the Fine-tuning Performance

To evaluate the fine-tuning performance over different number of source languages, we present a metric, dubbed average relative F1 score, which measures the average relative improvement of multi-lingual methods's fine-tuning performance comparing with the single-lingual fine-tuning. Specifically, let S_t^s be the F1 score of the single-lingual fine-tuning on language t, and let S_t^m be the F1 score of

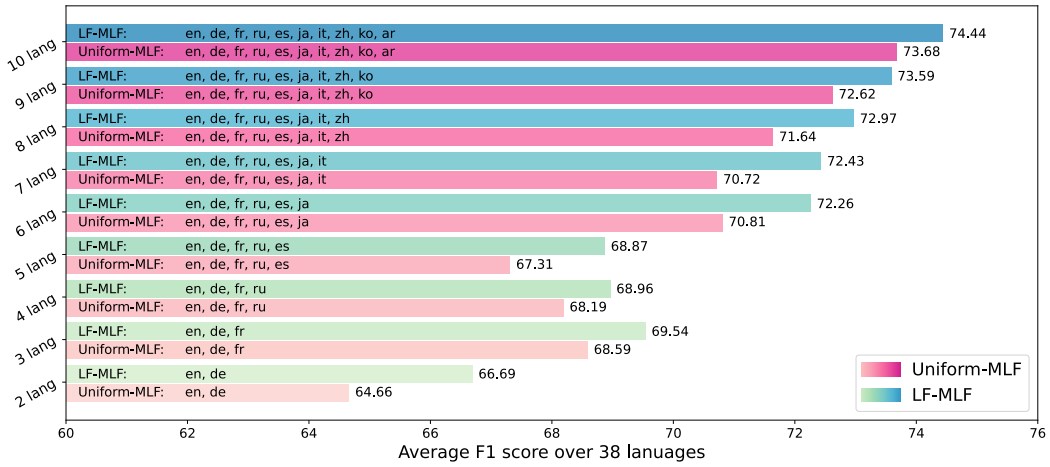


Figure 1: Zero-shot performance w.r.t different number of fine-tuning languages.

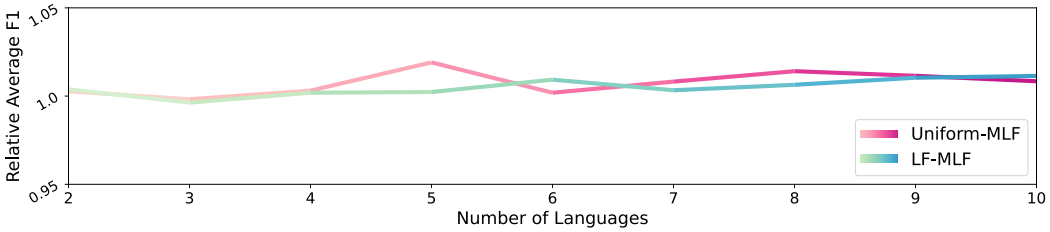


Figure 2: Relative fine-tuning performance w.r.t different number of fine-tuning languages.

the multi-lingual fine-tuning on language t. Assume there are T fine-tuning languages, the average relative F1 score is defined as $R_T = \frac{1}{T}\sum_{t=1}^{T}\frac{S_t^m}{S_t^s}$.

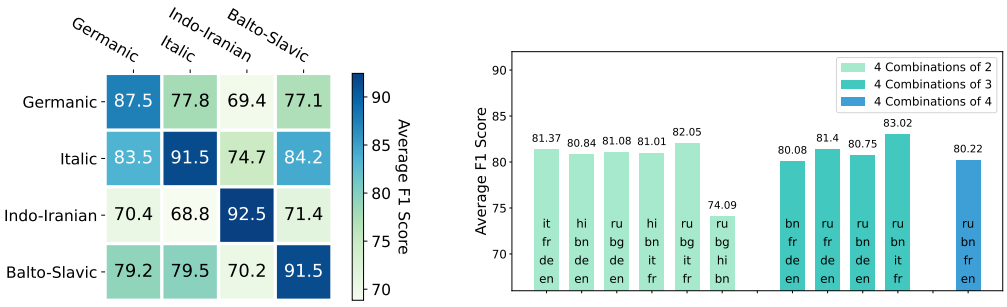
Figure 1 demonstrates the change of fine-tuning performance along with the increasing number of source languages. From this figure, we can see that (1) the fine-tuning performance just slightly fluctuates when the number of source languages changes; (2) LF-MLF and Uniform-MLF do not evidently outperform single-lingual fine-tuning, and they have similar average relative F1 score over different number of source languages.

5.4 Impact of the Diversity of Language Branches Covered by the Source Languages

After investigating the impact of the number of source languages, we further explore whether the diversity of language branches covered by the source languages impacts the zero-shot performance. In this section, we conduct experiments on the different branches of Indo-European language family based on the NER dataset. Specifically, we choose sixteen languages, which belong to four different language branches (Germanic, Italic, Indo-Iranian, Balto-Slavic) and each branch has four languages, from the NER dataset.

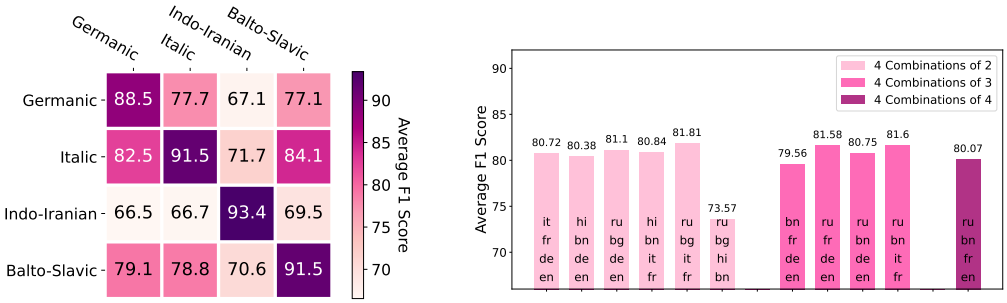
Firstly, we investigate the cross-language-branches transfer performance, namely fine-tuning on the four languages of a branch and observe the zero-shot performance on another branch, such as fine-tuning on Germanic and test zero-shot performance on Italic. The experimental results are illustrated in Fig. 3 (a) and Fig. 4 (a) for LF-MLF and Uniform-MLF respectively. These results present that: (1) the cross-language-branches transfer performance of each pair of branches is asymmetric; (2) LF-MLF has better cross-language-branches transfer ability than Uniform-MLF, which experimentally explains the superiority of LF-MLF and further verifies LF-MLF's less forgetting property.

Next, we explore the impact of language branches' diversity of the source languages. In this exploration, eight languages are used as candidates for fine-tuning (i.e., source languages), while the



(a) Cross Language Branches (b) Different Combinations of Language Families

Figure 3: Impact of the diversity on the fine-tuning languages' branches for LF-MLF.



(a) Cross Language Branches (b) Different Combinations of Language Families

Figure 4: Impact of the diversity on the fine-tuning languages' branches for Uniform-MLF.

other eight languages are used to test the zero-shot performance. The detailed allocation of these languages are reported in Table 7. Fixing the number of fine-tuning languages as four, we test the language branches' combinations that the source languages cover two branches, three branches and all four branches respectively, and the results are illustrated in Fig. 3 (b) and Fig. 4 (b). From these figures, we can see that: (1) the language branches' diversity of the fine-tuning languages does not have a clear impact on the zero-shot performance, which means that covering more language branches might not lead to better zero-shot performance; (2) LF-MLF generally outperforms Uniform-MLF on most of the combinations.

Language Family	Germanic	Italic	Indo-Iranian	Balto-Slavic
Fine-tuning Candidates	en, de	fr, it	bn, hi	bg, ru
Test Set for Zero-shot	af, nl	pt, es	mr, ur	pl, uk

Table 7: Data allocation for the language branches.

6 Conclusion

This paper researches on a novel fine-tuning setting for multi-lingual language models, namely multi-lingual fine-tuning; furthermore, we propose a novel multi-lingual fine-tuning method, dubbed Less-forgetting Multi-lingual Fine-tuning (LF-MLF). Comparing with its mono-lingual fine-tuning counterparts, multi-lingual fine-tuning can achieve superior zero-shot performance; moreover, our proposed LF-MLF outperforms the baseline multi-lingual fine-tuning methods. Researching on mono-lingual fine-tuning provides several inspirations and has the potential to forge new trends in multi-lingual Learning research.

References

[1] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Translation artifacts in cross-lingual transfer learning. In *EMNLP*, 2020.

[2] Sumanth Doddapaneni, Gowtham Ramesh, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh M. Khapra. A primer on pretrained multilingual language models. *CoRR*, 2021.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[4] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *ACL*, 2020.

[5] Zihan Liu, Genta Indra Winata, Andrea Madotto, and Pascale Fung. Exploring fine-tuning techniques for pre-trained cross-lingual models via continual learning. *CoRR*, 2020.

[6] Yuxuan Wang, Wanxiang Che, Jiang Guo, Yijia Liu, and Ting Liu. Cross-lingual BERT transformation for zero-shot dependency parsing. In *EMNLP-IJCNLP*, 2019.

[7] Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. Emerging cross-lingual structure in pretrained language models. In *ACL*, 2020.

[8] Zhiyuan Chen and Bing Liu. *Lifelong Machine Learning, Second Edition*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2018.

[9] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *NeurIPS*, 2017.

[10] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[11] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *NeurIPS*, 2017.

[12] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *NeurIPS*, 2018.

[13] Debabrata Mahapatra and Vaibhav Rajan. Multi-task learning with user preferences: Gradient descent with controlled ascent in pareto optimization. In *ICML*, 2020.

[14] Xi Lin, Zhiyuan Yang, Qingfu Zhang, and Sam Kwong. Controllable pareto multi-task learning. *CoRR*, 2020.

[15] Pingchuan Ma, Tao Du, and Wojciech Matusik. Efficient continuous pareto exploration in multi-task learning. In *ICML*, 2020.

[16] Yuren Mao, Shuang Yun, Weiwei Liu, and Bo Du. Tchebycheff procedure for multi-task text classification. In *ACL*, 2020.

[17] Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning. In *NeurIPS*, 2020.

[18] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *AISTATS*, 2020.

[19] James Kirkpatrick, Razvan Pascanu, Neil C. 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. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences of the United States of America*, 114(13):3521—-3526, 2017.

[20] M Schmidt, D Kim, and S Sra. of book: Optimization for machine learning. MIT Press, 2011.

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*. 2019.

[22] Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *ICML*, 2020.

[23] Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. Cross-lingual name tagging and linking for 282 languages. In *ACL*, 2017.

[24] Jonathan H. Clark, Jennimaria Palomaki, Vitaly Nikolaev, Eunsol Choi, Dan Garrette, Michael Collins, and Tom Kwiatkowski. Tydi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Trans. Assoc. Comput. Linguistics*, 8:454–470, 2020.

[25] Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. XNLI: evaluating cross-lingual sentence representations. In *EMNLP*, 2018.

[26] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *NeurIPS*, 2020.

[27] Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *ICLR*, 2021.

[28] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *CoRR*, 2019.

[29] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, 2017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [Yes]

(c) Did you include any new assets either in the supplemental material or as a URL? [No]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]