Unlocking the Potential of Model Merging for Low-Resource Languages

Mingxu Tao[1,3*], Chen Zhang[1*], Quzhe Huang[1*], Tianyao Ma[1]
Songfang Huang[4], Dongyan Zhao[1,2,3], Yansong Feng[1]✉
[1]Wangxuan Institute of Computer Technology, Peking University
[2]State Key Laboratory of General Artificial Intelligence, Peking University
[3]Center for Data Science, Peking University
[4]College of Engineering, Peking University
{thomastao,zhangch,huangquzhe,fengyansong}@pku.edu.cn

Abstract

Adapting large language models (LLMs) to new languages typically involves continual pre-training (CT) followed by supervised fine-tuning (SFT). However, this CT-then-SFT approach struggles with limited data in the context of low-resource languages, failing to balance language modeling and task-solving capabilities. We thus propose a new model merging solution as an alternative for low-resource languages, combining models with distinct capabilities into a single model without additional training. We use model merging to develop task-solving LLMs for low-resource languages without SFT data in the target languages. Our experiments based on `Llama-2-7B` demonstrate that model merging effectively endows LLMs for low-resource languages with task-solving abilities, outperforming CT-then-SFT in scenarios with extremely scarce data. Observing performance saturation in model merging with increasingly more training tokens, we further analyze the merging process and introduce a slack variable to the model merging algorithm to mitigate the loss of important parameters, thereby enhancing model performance. We hope that model merging can benefit more human languages suffering from data scarcity with its higher data efficiency.

1 Introduction

Large language models (LLMs) demonstrate remarkable capabilities across various NLP tasks, owing to the vast amounts of high-quality training data (Touvron et al., 2023; Bai et al., 2023). However, developing models with task-solving abilities for low-resource languages remains challenging due to limited data availability.

A common practice for constructing task-solving LLMs for a low-resource language involves continual pertaining (CT) and supervised fine-tuning (SFT) for the target language (Yong et al., 2023; Nguyen et al., 2023), known as **CT-then-SFT**. The scarcity of CT data impedes LLMs' ability to learn effective language modeling for these target languages. Additionally, it is difficult to acquire sufficient SFT data in low-resource languages to enhance downstream task performance. To address this issue, previous works attempt to transfer capabilities from high-resource languages to low-resource languages by training on English SFT data (Chirkova and Nikoulina, 2024; Shaham et al., 2024). However, this approach can lead to catastrophic forgetting (Thrun, 1998; Chen and Liu, 2018) of language modeling for the target languages (Mehta et al., 2021; Kotha et al., 2024), resulting in LLMs still failing to solve tasks due to the loss of language abilities.

To better integrate the language modeling and task-solving capabilities, we introduce model merging for low-resource languages, which can combine multiple models with distinct abilities into a single model without additional training. Previous work (Akiba et al., 2024) has shown that an LLM for high-resource languages can be merged with task-specific models, such as Japanese language models and math models. In this work, we explore whether model merging can effectively construct task-solving LLMs for low-resource languages. Specifically, we investigate the following research questions: **RQ1:** What is the viability of constructing task-solving LLMs in low-resource languages through model merging? **RQ2:** Is model merging always a better choice than CT-SFT? **RQ3:** What factors may affect LLMs in obtaining task-solving capabilities through model merging?

To answer these questions, we study the adaptation of Llama-2-7B (Touvron et al., 2023), an English-centric LLM, into seven distinct low-resource languages. We first continually pre-train Llama-2-7B on monolingual texts in each language. Next, we explore two approaches to inject task-solving capabilities into this continually pre-trained

*Equal contributions.

Findings of the Association for Computational Linguistics: EACL 2024, pages 8705–8720
November 12-16, 2024

model: (1) training the LLM with English SFT data or the data translated to the target low-resource language; (2) merging the model with an English task-solving LLM. Experiments show that model merging can effectively equip the CT models with task-solving capabilities. Notably, when pre-training corpora in the target language are extremely scarce (<10B tokens), model merging outperforms CT-then-SFT. As an LLM is continually pre-trained with more tokens in target languages, the improvements brought by model merging gradually saturate. Then, model merging can no longer significantly surpass the SFT method.

To further investigate the factors impeding the continuous improvement of model merging, we conduct a detailed analysis of the process of merging two LLMs. We find that when an English SFT model is merged with an LLM continually pre-trained with more tokens in the target language, more parameters from the SFT model are discarded during merging. The loss of these parameters may lead to a decline in task-solving capabilities, preventing the merged model from improving performance on downstream tasks. To mitigate the loss of important parameters from the SFT model, we propose *a novel model merging solution with slack variables*. This strategy allows for more flexible control over the merging process to retain important parameters.

Our contributions are as follows: (1) We are the first to introduce model merging to construct task-solving LLMs for low-resource languages; (2) We reveal that model merging is more effective than SFT in the scenarios of extremely low-resource languages; (3) Through a quantitative study of the merging process, we explain the performance plateau of model merging with a larger CT corpus and propose a simple yet effective enhancement to popular model merging algorithms.

2 Related Works

Model Merging Model merging is a promising way to combine the abilities of multiple models. Pioneering works explore strategies to find the best weights for averaging (Choshen et al., 2022; Wortsman et al., 2022; Matena and Raffel, 2022; Jin et al., 2022). Task Arithmetic (Ilharco et al., 2022) employs task vectors, enabling control through arithmetic operations to steer the merged model's behavior. TIES (Yadav et al., 2023) further addresses the problem of information loss by handling parameter conflict more carefully. DARE (Yu et al., 2023) zeros out redundant parameters and amplifies the remaining ones. Evolutionary Model Merge (Akiba et al., 2024) automatically discovers optimal model combinations through evolutionary algorithms.

There is little discussion of model merging in the context of multilinguality. Instead, previous works attempt to introduce language-specific and task-specific modular adapters (Pfeiffer et al., 2020; Parovic et al., 2023; Parović et al., 2024; Zhao et al., 2024), which require additional training. These works focus on high-resource languages and specific tasks. In contrast, model merging can utilize existing models without additional training, making it a versatile approach for building more general task-solving LLMs. Besides, we are the first to study model merging for low-resource languages.

LLMs for Low-Resource Languages There is a line of works aiming to adapt LLMs to under-represented human languages. A common practice is continually pre-training existing LLMs on the corpus in the target languages (Yong et al., 2023; Nguyen et al., 2023; Zhang et al., 2024b). To improve the efficiency of training, previous works adopt techniques such as adapters (Pfeiffer et al., 2020), script conversion (Micallef et al., 2024), integration of similar languages (Senel et al., 2024).

Following pre-training, LLMs typically undergo supervised fine-tuning to acquire task-solving capabilities (Muennighoff et al., 2023; Nguyen et al., 2023). To address the data scarcity in this step, researchers have employed various methods to collect SFT data, including crowd-sourcing (Singh et al., 2024), machine translation (Muennighoff et al., 2023; Li et al., 2023a), LLM distillation (Li et al., 2024) ,rule-based conversion (Cahyawijaya et al., 2023), et al. However, these methods are not without limitations, particularly in terms of cost, data quality, and generalizability. In contrast, the model merging paradigm studied in our work eliminates the need for expensive and potentially error-prone SFT data collection by leveraging pre-trained task-solving models from high-resource languages.

3 Model Merging for Low-Resource Languages

The conventional CT-then-SFT paradigm struggles to balance language modeling and task-solving abilities in the context of low-resource languages. We propose to adopt model merging as an alternative, which can construct task-solving LLMs for low-

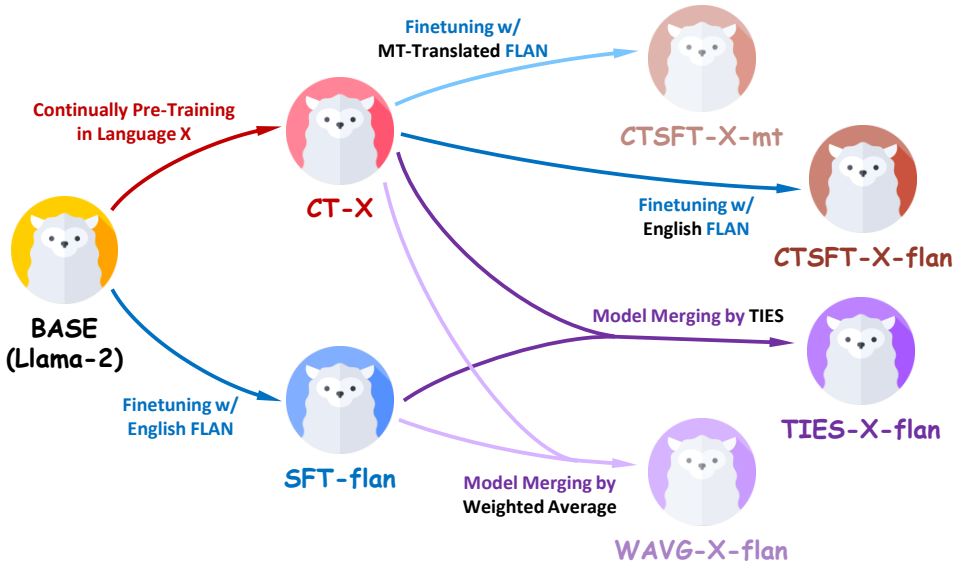

Figure 1: Roadmap towards task-solving LLMs for low-resource languages.

resource languages without requiring SFT data in the target languages.

3.1 Preliminary: Model Merging

Model merging is a technique for combining multiple models possessing different capabilities into a single versatile model without additional training. For example, we can merge a model specialized for Japanese and a model specialized for math to obtain a model that excels at solving mathematical problems in Japanese (Akiba et al., 2024). In this work, we investigate two commonly-used methods of model merging: **weighted averaging** (Choshen et al., 2022; Wortsman et al., 2022) and **TIES** (Yadav et al., 2023). Here we provide a brief overview of these methods.

Weighted averaging is simply averaging the parameters of two models with a weight tuned on the validation set.

TIES aims to handle the parameter conflicts across multiple models more meticulously. Suppose we have two models specialized for distinct tasks, denoted as θ_1 and θ_2, both trained from the same initial model θ_{init}. *Task vectors* for these models are calculated as follows: $\tau_1 = \theta_1 - \theta_{\text{init}}$ and $\tau_2 = \theta_2 - \theta_{\text{init}}$. The objective is to merge these task vectors and reintegrate them into the initial model.

The merging process of TIES consists of three steps: (1) **Trim**: For τ_1 and τ_2, we trim the redundant parameters by keeping the top-k_1% and top-k_2% values, respectively, creating $\hat{\tau}_1$ and $\hat{\tau}_2$. (2) **Elect Signs**: For each parameter p in $\hat{\tau}_1$ and $\hat{\tau}_2$, we select the sign (+1 or -1) with the higher magnitude, denoted as $\gamma^p = \text{sgn}(\hat{\tau}_1^p + \hat{\tau}_2^p)$. (3) **Disjoint Merge**: For each parameter p, we only keep the parameter values from $\hat{\tau}_1$ and $\hat{\tau}_2$ whose signs are the same as the aggregated elected sign and calculate their mean. Specifically, for each parameter p, its disjoint mean is calculated as $\tau_m^p = \text{avg}(S^p)$, where $S^p = \{\hat{\tau}_i^p | \text{sgn}(\hat{\tau}_i^p) = \gamma^p, i = 1, 2\}$.

Given the final merged task vector τ_m, we scale it and add it to the initial model θ_{init} to obtain the merged model θ_m as $\theta_m = \theta_{\text{init}} + \lambda \cdot \tau_m$, where λ is a scaling hyperparameter.

For TIES, we tune three hyperparameters in total on the validation set: two sparsity rates k_1, k_2 and a scaling factor λ.

Please refer to the original paper of TIES (Yadav et al., 2023) for more details.

3.2 Roadmap Towards LLMs for Low-Resource Languages

Given a base model only pre-trained on an English-centric corpus, e.g., Llama-2-7B (Touvron et al., 2023) in our study, we want to construct a model capable of solving tasks in a low-resource language. For those target languages, there are usually very limited pre-training texts, ranging from 1B to 20B tokens, and almost no data for supervised finetuning (SFT). In this scenario, we investigate two representative paradigms of constructing such a model: CT-then-SFT and model merging. We illustrate the roadmap in Figure 1, which demonstrates the relations among the models.

Conventional Practice: CT-then-SFT The common practice is (1) first continual pre-training (CT) on the monolingual texts in the target language `X` to learn the language modeling and (2) then learning task-solving abilities through SFT (Yong et al., 2023; Nguyen et al., 2023). This approach is referred to as **CT-then-SFT**. Specifically, we consider the following models:

BASE: We employ the original Llama-2-7B without SFT as the base LLM.

CT-`X`: We continually pre-train BASE on the corpus in the target language `X`. Following previous works (d'Autume et al., 2019; Tao et al., 2023), we add 1/4 English corpus for memory replay, to avoid catastrophic forgetting English language modeling.

CTSFT-`X`: We train CT-`X` with SFT data to enhance its task solving ability. There are two variants using different SFT data:

(1) **CTSFT-`X-flan`**: We finetune CT-`X` with English SFT data, which includes the original FLAN datasets and the training set of GSM8K[1]. This approach is based on the assumption that task-solving abilities in English can be transferred to the target language (Chirkova and Nikoulina, 2024; Shaham et al., 2024).

(2) **CTSFT-`X-mt`**: We translate FLAN and the training set of GSM8K into the target language `X` with machine translation (MT) systems[2], which is a common practice to obtain SFT data for non-English languages (Muennighoff et al., 2023; Li et al., 2023a,b). We then finetune CT-`X` with the obtained SFT data.

New Paradigm: Model Merging By model merging, we can integrate distinct LLMs with various capabilities into one LLM. To obtain a model capable of solving tasks in the target language `X`, we can merge the following two models:

CT-`X`: As discussed in the CT-then-SFT procedure, this model learns a certain amount of language modeling in the language `X` during CT. However, its task-solving ability is limited.

SFT-`flan`: We directly finetune BASE with the SFT data used by CTSFT-`X-flan`. The resulting model has sufficiently learned task solving, but the target language `X` is still foreign to it.

We merge the two models above to unlock the dual benefits of proficient language modeling and effective task-solving capabilities. Specifically, we investigate two methods of model merging: **weighted averaging** (WAVG, Choshen et al., 2022; Wortsman et al., 2022) and **TIES** (Yadav et al., 2023). We derive two variants of merged models, namely **WAVG-`X-flan`** and **TIES-`X-flan`**.

4 Experimental Setup

Languages We use 7 low-resource languages from five distinct language families for experiments: Tamil, Telugu, Odia, Bengali, Tibetan, Uyghur, and Mongolian (in the traditional Mongolian script). See their basic information in Table 1.

We select these languages because they are underrepresented in currently popular LLMs despite their large population (over 475M) worldwide. As shown in Table 3, the performance of Llama-2-7B in these languages is close to or even worse than random guessing[3]. Notably, the vocabulary of Llama-2 does not even contain tokens for Odia and traditional Mongolian, which indicates that the model has hardly seen these languages during pre-training. Moreover, limited resources are available for these languages on the internet. Among those languages, we can only collect fewer than 1B tokens of monolingual texts for Odia, Tibetan, Uyghur, and traditional Mongolian. The problem of data scarcity becomes more severe in terms of high-quality data for supervised fine-tuning.

Pre-training Corpus During continual pre-training, we use the largest available corpus for each language from CulturaX (Nguyen et al., 2024), IndicCorp-v2 (Doddapaneni et al., 2023), and MC^2 (Zhang et al., 2024b). To maximize language coverage with constraint computational resources, we sample 8B tokens for continual pre-training of Tamil and Telugu, and 16B tokens for Bengali. The corpus sizes are shown in Table 2.

Following Llama models (Touvron et al., 2023), we employ RedPajama (Computer, 2023) with the same sampling proportion for memory replay to reduce forgetting of English language modeling.

SFT Data We mainly use FLAN (Longpre et al., 2023) for SFT, which consists 155K training instances for 1,411 distinct tasks. Since there are limited math reasoning tasks in FLAN, we additionally

[1]Since the whole instruct-tuning datasets contain over 160K instances, we perform necessary replay with pre-training texts in both English and language `X`.

[2]We use NLLB (NLLB Team et al., 2022) for translation. To enhance the model's ability to follow English prompts, we randomly translate half of the training instances into language `X`, while the other half of instances remain in English.

[3]The accuracy of random guessing should be 25% for Belebele, 14.29% for SIB-200, and 25% for the multiple-choice tasks in the MLiC-Eval benchmark.

Name	Family	Script	Population
Tamil (tam)	Dravidian	Tamil	79M
Telugu (tel)	Dravidian	Telugu	96M
Odia (ory)	Indo-Euro.	Odia	35M
Bengali (ben)	Indo-Euro.	Bengali	240M
Tibetan (bod)	Sino-Tibetan	Tibetan	7M
Uyghur (uig)	Turkic	Arabic	12M
Mongolian (mvf)	Mongolic	Mongolian	6M

Table 1: Languages families, writing systems, and populations of the low-resource languages in our study.

Lang.	Corpus	Tokens	Tasks
tam	CulturaX	15.9B	SIB-200, Belebele
tel	CulturaX	12.4B	SIB-200, Belebele, MGSM
ory	CulturaX	765M	SIB-200, Belebele
ben	IndicCorp-v2	36.4B	SIB-200, Belebele, MGSM
bod	MC^2	1.00B	MLiC-Eval
uig	MC^2	412M	MLiC-Eval
mvf	MC^2	904M	MLiC-Eval

Table 2: The pre-training corpus and evaluation tasks of each language in our study. The number of tokens is obtained with the tokenizer of Llama-2.

incorporate 7,473 instances from GSM8K (Cobbe et al., 2021) into the supervised training sets.

We translate FLAN into above mentioned languages using NLLB-200-Distilled-1.3B (NLLB Team et al., 2022)[4]. Note that this model does not support traditional Mongolian and there are no open-source MT models available for this language currently. We thus adopt a roundabout way: translating the instructions into Cyrillic Mongolian, which NLLB supports, and converting them into traditional Mongolian with an open-source transliteration tool[5].

Evaluation Tasks Regarding the four languages in India (tam, tel, ory, and ben), we use SIB-200 (Adelani et al., 2024) for text classification, Belebele (Bandarkar et al., 2023) for machine reading comprehension, and MGSM (Shi et al., 2022) for math reasoning (only available in Telugu and Bengali). Regarding the three languages in China (bod, uig, and mvf), we use MLiC-Eval (Zhang et al., 2024a), including the following 4 tasks: text classification (TC), machine reading comprehension (MRC), response selection (RS), and math reasoning. See statistics in Appendix A.

Implementation Details We use Megatron-LM for model training (Shoeybi et al., 2019) and Arcee's MergeKit for model merging (Goddard et al., 2024). See more details in Appendix B.

5 Results and Analysis

In this work, we mainly study two categories of common roadmaps, *CT-then-SFT* and *model merging*, to transfer the task-solving ability from a high-resource language to a low-resource one. We aim to investigate following research questions. **RQ1:** What is the viability of constructing task-solving LLMs in low-resource languages via model merging? **RQ2:** Is model merging always a better choice than CT-then-SFT?

We take the settings of BASE, SFT-`flan`, and CT-`X` as the baselines. For *CT-then-SFT*, we investigate two common methods to build task-solving models, CTSFT-`X-flan` and CTSFT-`X-mt`. For *model merging*, we study two effective algorithms to combine the abilities of language modeling and task solving: weighted averaging and TIES.

Table 3 illustrates the overall performance of different models or setups, i.e., the average scores over all tasks, for each language. For the experiments involving continual pre-training, we report the results based on the last checkpoints of CT-`X`[6]. See the model performance on individual tasks in Appendix C.1.

5.1 Effectiveness of Model Merging

For all the studied languages except Bengali, the models constructed by *merging* outperform those built by *CT-then-SFT*[7]. For example, the merged models based on TIES (TIES-`X-flan`) achieve an average score of 46.80% across languages, surpassing CT-then-SFT models (CTSFT-`X-flan`) by +4.69%. Although using a naïve merging algorithm, model merging with WAVG (WAVG-`X-flan`) still achieves better performance than CT-then-SFT (CTSFT-`X-flan`) in four languages.

In the conventional *CT-then-SFT* approach, LLMs often fail to acquire sufficient comprehension of the target language from a small-size CT corpus, and this ability may be further diminished by supervised fine-tuning. In contrast, *model merging* can preserve the language modeling acquired

[4]NLLB is currently the open-source MT model with the most extensive language support.

[5]https://github.com/tugstugi/mongolian-nlp/tree/master/bichig2cyrillic

[6]Due to the budget of computational resources and available pre-training data, we use 8B tokens for continual pre-training on Tamil and Telugu respectively, and 16B tokens for continual pre-training on Bengali.

[7]The exceptional performance in Bengali may be attributed to its larger CT corpus, which we discuss in Section 5.2.

	Task	Lang.	Tamil	Telugu	Odia	Bengali	Tibetan	Uyghur	Mongolian	Average
BASE			28.15	18.83	26.64	25.47	13.49	13.34	11.57	19.64
SFT-flan	✓		29.19	17.28	25.21	24.84	23.29	22.27	19.86	23.13
CT-X		✓	52.18	34.67	47.93	30.77	13.52	14.80	11.09	29.28
CTSFT-X-mt	✓	✓	50.57	32.90	30.14	38.40	33.85	24.85	19.57	32.90
CTSFT-X-flan	✓	✓	53.95	37.96	44.56	**42.19**	42.36	49.46	24.29	42.11
WAVG-X-flan	✓	✓	57.56	37.58	53.59	37.19	44.30	42.64	31.09	43.42
TIES-X-flan	✓	✓	**58.46**	**39.50**	**56.49**	40.31	**47.86**	**52.43**	**32.56**	**46.80**

Table 3: Performance of models built through the roadmap. The best results are made **bold**, with the second underlined. The **Task** and **Lang.** columns denote whether the model have enhanced task-solving abilities and learned the target languages, respectively.

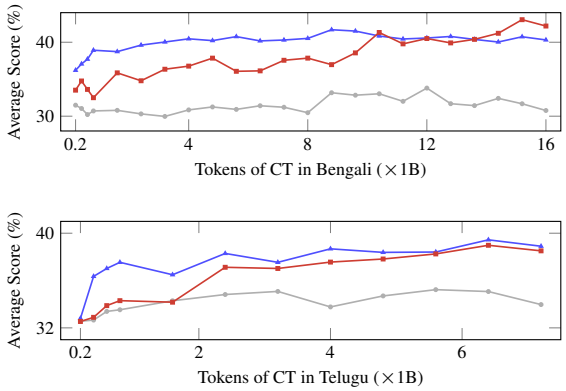

Figure 2: Performance of the models based on each checkpoint of CT-ben and CT-tel. The blue lines illustrate results of TIES-X-flan, with red lines for CTSFT-X-flan and grey lines for CT-X.

during CT, even with a small-size CT corpus, while incorporating task-solving capabilities by resolving parameter conflicts carefully.

In conclusion, **model merging can be an effective pathway to obtain task-solving LLMs for low-resource languages**.

5.2 Performance Plateau of Model Merging

Although *model merging* is shown to be more effective than *CT-then-SFT* in almost all languages of our study, this is not the case for Bengali, which has the largest CT corpus in our experiments: CTSFT-ben-flan outperforms TIES-ben-flan. We guess that the applicability of model merging may be related to the amount of corpus used in CT. Thus, we examine the performance changes of the merged models and the CT-then-SFT models under different amounts of CT tokens.

We collect the intermediate checkpoints of CT-ben and CT-tel, which are the two languages with the largest sizes of pre-training corpora. Then, we derive CTSFT-X-flan and TIES-X-flan models based on these checkpoints.

Figure 2 illustrates the performance of CT-only (CT-X), CT-then-SFT (CTSFT-X-flan), and model merging (TIES-X-flan) based on every checkpoint. For Bengali and Telugu, TIES-X-flan outperforms CT-X in each checkpoint, indicating model merging can robustly enhance LLM's task-solving ability, while *CT-then-SFT* shows greater variability. And in both languages, for all checkpoints where the amount of pre-trained tokens is less than 10.4B tokens, TIES-X-flan can achieve better results than CTSFT-X-flan.

Model merging can integrate language modeling and task-solving capabilities more effectively than CT-then-SFT, in scenarios with limited language resources (<10B tokens in our experiments). Four out of seven languages in our study have pre-training corpora with fewer than 10B tokens. This data scarcity is intrinsic for low-resource languages. For instance, 81% (135 out of 166) of the languages in the multilingual corpus CulturaX (Nguyen et al., 2024) have fewer than 10B tokens. Consequently, our findings on the effectiveness of model merging are broadly applicable and have the potential to benefit a wide range of human languages.

As the amount of CT tokens increases, the CTSFT-X-flan models show more rapid improvement in task-solving capabilities compared to TIES-X-flan. Specifically, when pre-trained with more than 14.4B tokens, CTSFT-ben-flan demonstrates better performance over TIES-ben-flan. Similarly, in Telugu, the performance gap between the two models slightly diminishes with additional CT tokens. However, due to the smaller size of the Telugu corpus compared to Bengali's, we have not observed CTSFT-tel-flan overtaking TIES-tel-flan in our experiments. We note that **CT-then-SFT may be a better method to construct task-solving LLMs in languages with sufficient resources, for example, Bengali, Vietnamese, In-**

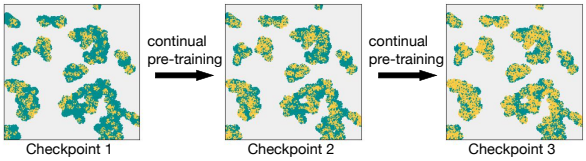

Figure 3: Diagram illustrating the change in the proportion of SFT-flan parameters discarded during sign election as CT progresses. The colored areas represent the parameters with sign conflicts. Among them, the cyan parts represent the parameters which elect the signs of SFT-flan, while the yellow parts for parameters selecting the signs of CT-ben.

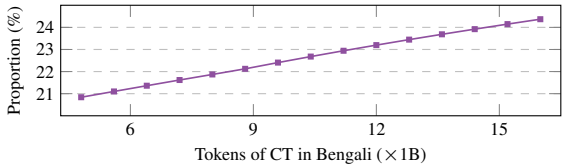

Figure 4: The proportion of parameters discarded in SFT-flan during sign election, relative to the total number of retained parameters of SFT-flan after trimming.

donesian, etc.

6 Understanding the Dynamics of Model Merging

As shown in Figure 2, we find that the performance of TIES-X-flan on downstream tasks may no longer improve as we use more tokens for CT. In this section, we want to investigate **RQ3:** What factors may affect LLMs in obtaining task-solving capabilities through model merging?

6.1 Quantifying Parameter Conflicts

Revisiting the mechanism of TIES, we find that during the merging stage, parameters from one model may be discarded due to differences in the parameter signs between the two models. As we continually pre-train the LLM with more tokens in language X, the parameters of CT-X changes more significantly. Since the magnitude of CT-X's *task vector* becomes larger, more parameters of SFT-flan would be discarded during the process of electing signs.Figure 3 illustrates the changes of discarded parameters in SFT-flan.

The discarding of model parameters usually leads to a decline in the corresponding capabilities. Regarding that the CT model's language modeling ability in language X continuously improves with the increase of pre-training data, we suspect that it is more likely that the SFT model is forced to discard too much information during the merging process, resulting in the merged model's inability to further enhance its task-solving ability in the target language.

To verify this hypothesis, we take Bengali, the language with the largest amount of CT corpus in our study, as an example. We track the changes in the number of discarded parameters in the SFT model when merging it with CT models trained with different amounts of data.

As explained in Section 3.1, we first calculate the *task vectors* of SFT-flan and each checkpoint of CT-ben. In the trimming stage, we find the optimal hyperparameters are $k_{\text{sft}} = 0.2$ and $k_{\text{ct}} = 1.0$ for most scenarios[8]. Thus, to provide a fair comparison, we freeze $(k_{\text{sft}}, k_{\text{ct}})$ as $(0.2, 1.0)$ for all checkpoints. Then, we examine the signs and magnitudes of each parameter of the trimmed task vectors of SFT-flan and CT-ben. A parameter that has contrary signs in two models can be regarded as a parameter with sign conflict. And if its magnitude in SFT-flan is smaller than that in CT-ben, it will be removed at the stage of sign election.

Figure 4 illustrates the proportion of parameters that are discarded from SFT-flan during the merging stage. We can find as the LLM is pre-trained with more tokens, 4% more parameters are removed in trimmed SFT-flan. We believe that **when using more tokens for CT, TIES discards a larger proportion of parameters from the SFT model, which may continuously undermine the merged model's task-solving capabilities .**

6.2 Model Merging with a Slack Variable

To mitigate the information loss in SFT-flan during the merging stage, we design a new approach, **TIES-SV**, enhancing **TIES** with a **S**lack **V**ariable to carefully reduce the number of discarded parameters in the model with higher information density, i.e., SFT-flan in this situation.

According to the process of TIES, for each parameter to be discarded from SFT-flan in the Disjoint Merge step, its magnitude is smaller than the magnitude of its counterpart parameter in CT-X. To retain the parameters of SFT-flan while minimizing the information loss of CT-X, we first rank these pairs of parameters between SFT-flan and CT-X according to their differences in the magnitude. Next, we select a subset of parameters with the smallest magnitude differences to reserve.

[8]If not being optimal, this set of hyperparameters can also result in comparable performance to the optimal ones.

	MGSM	SIB-200	Belebele	Average
TIES	7.50	79.41	34.02	40.31
TIES-SV	**8.00**	**79.90**	**34.58**	**40.83**

Table 4: Results of TIES and TIES-SV on Bengali tasks.

We evaluate the effectiveness of TIES-SV on the model merging process of SFT-`flan` and the last checkpoint of CT-ben. Based on the results in Figure 4, we reserve 4% parameters in SFT-`flan` that were to be discarded in the original algorithm of TIES. Table 4 shows the results of vanilla TIES and our TIES-SV on three Bengali tasks. In all three tasks, our TIES-SV outperforms vanilla TIES by 0.52% on average.

It is surprising that this simple and intuitive strategy can bring such an improvement for the merged model, suggesting that different models' parameters may have different importance during the process of model merging. **When parameter conflicts occur, we cannot simply rely on the magnitude of the vectors to decide which models' parameters should be reserve. Instead, one should use prior information obtained through pilot studies or other means to reserve the parameters of the more important model.** We hope our TIES-SV can shed light on the study of new model-merging algorithms in the future.

7 Discussions

In this section, we further explore two important questions related to model merging. First, we investigate the potential of merging more than one low-resource language into a task-solving model, which could offer a new avenue for constructing multilingual models. Second, we examine why commonly-used machine-translated SFT data often fails in the context of constructing task-solving models in low-resource languages. This failure underscores the advantage of model merging, as it does not require SFT data in the target language.

7.1 Can We Merge Multiple Languages?

Previous work (Akiba et al., 2024) shows that model merging algorithms can be used to combine multiple LLMs with different capabilities, which may enhance the model to solve complex problems. We wonder whether multiple LLMs adapted to different low-resource languages can be merged with the same SFT model, to construct a task-solving LLM supporting these languages simultaneously.

Model	Ability	mvf	uig
SFT-`flan`	*A*	19.86	22.27
CT-`mvf`	*B*	11.09	13.58
CT-`uig`	*C*	11.65	14.80
TIES-`mvf-flan`	*A*+*B*	**32.56**	27.24
TIES-`uig-flan`	*A*+*C*	24.45	**52.43**
TIES-`mvf&uig-flan`	*A*+*B*+*C*	<u>30.61</u>	<u>52.40</u>

Table 5: Results of merging multiple languages into a task-solving model. The columns **mvf** and **uig** refer to the average performance across the Mongolian and Uyghur tasks, respectively. The best results are made **bold**, with the second <u>underlined</u>.

As a pilot study, we attempt to merge Mongolian and Uyghur LLMs with the English task-solving model SFT-`flan`. Due to the limited budget for computational resources, we cannot conduct a grid search across the hyperparameter space of the three models. Therefore, we employ the optimal hyperparameters derived from merging each of the two CT models with SFT-`flan`.

Table 5 illustrates the average scores of tasks in the two languages. The merged model serving two low-resource languages (TIES-`mvf&uig-flan`) performs comparably to the two single-language merged models (TIES-`mvf-flan` and TIES-`uig-flan`) on tasks in the respective languages.

This indicates that **model merging has great potential for constructing multilingual task-solving LLMs**. We hope this approach can assist multilingual speakers, particularly those using underrepresented languages, by combining multiple existing LLMs in distinct languages without the need for expensive pre-training.

7.2 Why do Machine Translated Data Fail?

Collecting synthetic SFT data through MT is an intuitive method for constructing task-solving LLMs in non-English languages (Muennighoff et al., 2023; Li et al., 2023a). However, the experimental results in Table 3 show that MT-translated SFT data may not work when it comes to low-resource languages. The models trained with MT-translated data (CTSFT-`X-mt`) exhibit inferior performance across all languages compared to those trained on English SFT data (CTSFT-`X-flan`), with a gap of -3.38%∼-24.61%.

The decline in model performance can be attributed to the low-quality MT results. Current open-source MT systems have limited abilities for low-resource languages. For example, we ask

Uyghur and Tibetan native speakers to evaluate sampled translation results by NLLB. They find that there are often irrelevant contents and undesired code-switching in the translation results, as shown in Appendix C.2. This kind of noise not only hinders the model's ability to learn task-solving strategies but also interferes with language modeling, which itself is inadequately learned.

In contrast, model merging eliminates the need to collect SFT data in the target languages. This approach also eliminates the reliance on unreliable machine translation systems.

8 Case Study

We further investigate how model merging can improve the model performance on the tasks in low-resource languages. Here, we focus on two types of errors in low-resource language models: **unparsable outputs during generation**, and **misunderstanding of character overlap** in MRC. In this section, we take traditional Mongolian as an example, which the BASE model has hardly seen during pre-training.

8.1 Unparsable Output

Note that the tokenizer of Llama-2 does not contain any tokens in the traditional Mongolian script. Therefore, the Mongolian texts should be encoded as byte tokens. For example, the Mongolian character "ᠪ" is tokenized as `[0xe1, 0xa0, 0xa4]`. During the inference process, the model generates byte by byte, which can be converted into Mongolian characters by UTF-8.

We find that in the Math task of MLiC-Eval, the models sometimes output incorrect byte combinations. These bytes cannot be converted into meaningful UTF-8 characters (displayed as ��). Furthermore, after such errors, the models continue to produce meaningless bytes and fail to provide the final answers.

We find CTSFT-`mvf-flan` produces meaningless bytes in 74 out of 200 instances, whereas TIES-`mvf-flan` outputs unparsable bytes in only 42 instances. Compared to the CT-then-SFT models, **the merged models have better abilities to generate texts in low-resource languages.**

8.2 Misunderstanding of Character Overlap

We also present a case study of the machine reading comprehension task. Here is the English translation of an instance in the MRC task of MLiC-Eval:

Context:
Man: Is the train station far from here?
Woman: It only take 20 minutes by taxi.
Question: What does the woman want to express?
Choices: A. Don't worry. B. It is not too far.

In this instance, the model needs to infer that a 20-minute taxi ride is not too far. TIES-`mvf-flan` can correctly answer this question. However, CTSFT-`mvf-flan` chooses the Choice A. This may be due to the character overlap between the phrases "taking a taxi (ᠲᠠᠺᠰᠢ ᠤᠮᠳᠠᠭᠠᠬᠤ)" and "don't worry (ᠪᠤᠷᠤᠭᠤᠳᠠᠬᠤ ᠦᠬᠡᠢ)" in traditional Mongolian.

The CT-then-SFT model demonstrates worse reasoning capabilities and tends to predict based on character overlap. We guess that fine-tuning on English datasets might primarily obtain the capabilities to handle English tasks but fail to transfer them to the low-resource languages.

9 Conclusion

In this paper, we investigate the potential of using model merging to construct task-solving LLMs for low-resource languages. Our findings demonstrate that model merging outperforms the conventional CT-then-SFT paradigm, achieving higher data efficiency. We further analyze the mechanism behind the performance saturation of model merging with an increased number of CT tokens, which inspires a simple yet effective improvement to the model merging algorithm. We hope that model merging can reduce the costs associated with data collection and model training, benefiting a greater number of languages suffering from data scarcity.

Limitations

Studied Languages Due to the high computational cost of continually pre-training LLMs, we cannot cover a wider range of low-resource languages, only focusing on seven underrepresented languages in India and China. However, we make efforts to improve the diversity of selected languages by including different language families and writing systems.

But we have to admit that, for these languages, we can obtain enough language resources to continually per-train LLMs with 7B parameters. We still cannot know whether CT-then-SFT and model merging can be effective to build task-solving models in the extremely low-resource languages. We believe it is important to explore the optimal method to build task-solving LLMs in a wider range of languages.

Evaluation Tasks Most of our evaluated tasks focus on natural language understanding, with less emphasis on natural language generation. This limitation arises from the insufficient CT corpus available for the studied low-resource languages, which is insufficient for the models to learn to perform complex generation in the target language.

Model Merging Algorithms We primarily discuss one popular model merging approach, TIES (Yadav et al., 2024), in this work. TIES is one of the most effective and efficient methods of model merging yet, which also yields optimal performance in previous work (Akiba et al., 2024) and does not need to access the training data during merging. We hope our insights can inspire more studies on other model merging methods in the context of building task-solving LLMs in low-resource languages.

Acknowledgements

This work is supported in part by NSFC (62161160339) and Beijing Science and Technology Program (Z231100007423011). We thank the anonymous reviewers for their helpful discussions and suggestions. For any correspondence, please contact Yansong Feng.

References

David Adelani, Hannah Liu, Xiaoyu Shen, Nikita Vassilyev, Jesujoba Alabi, Yanke Mao, Haonan Gao, and En-Shiun Lee. 2024. SIB-200: A simple, inclusive, and big evaluation dataset for topic classification in 200+ languages and dialects. In Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 226–245, St. Julian's, Malta. Association for Computational Linguistics.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. 2024. Evolutionary optimization of model merging recipes. arXiv preprint arXiv:2403.13187.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. arXiv preprint arXiv:2309.16609.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2023. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. arXiv preprint arXiv:2308.16884.

Samuel Cahyawijaya, Holy Lovenia, Tiezheng Yu, Willy Chung, and Pascale Fung. 2023. InstructAlign: High-and-low resource language alignment via continual crosslingual instruction tuning. In Proceedings of the First Workshop in South East Asian Language Processing, pages 55–78, Nusa Dua, Bali, Indonesia. Association for Computational Linguistics.

Zhiyuan Chen and Bing Liu. 2018. Lifelong supervised learning. In Ronald J. Brachman and Peter Stone, editors, Lifelong Machine Learing, 2nd edition, pages 35 – 54. Morgan & Claypool Publishers.

Nadezhda Chirkova and Vassilina Nikoulina. 2024. Zero-shot cross-lingual transfer in instruction tuning of large language model. arXiv preprint arXiv:2402.14778.

Leshem Choshen, Elad Venezian, Noam Slonim, and Yoav Katz. 2022. Fusing finetuned models for better pretraining. arXiv preprint arXiv:2204.03044.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168.

Together Computer. 2023. Redpajama: An open source recipe to reproduce llama training dataset.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, pages 13132–13141.

Sumanth Doddapaneni, Rahul Aralikatte, Gowtham Ramesh, Shreya Goyal, Mitesh M. Khapra, Anoop Kunchukuttan, and Pratyush Kumar. 2023. Towards leaving no Indic language behind: Building monolingual corpora, benchmark and models for Indic languages. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 12402–12426, Toronto, Canada. Association for Computational Linguistics.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's mergekit: A toolkit for merging large language models. arXiv preprint arXiv:2403.13257.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. In The Eleventh International Conference on Learning Representations.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2022. Dataless knowledge fusion by merging weights of language models. In The Eleventh International Conference on Learning Representations.

Suhas Kotha, Jacob Mitchell Springer, and Aditi Raghunathan. 2024. Understanding catastrophic forgetting in language models via implicit inference. In The Twelfth International Conference on Learning Representations.

Chong Li, Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2023a. Align after pre-train: Improving multilingual generative models with cross-lingual alignment. arXiv preprint arXiv:2311.08089.

Chong Li, Wen Yang, Jiajun Zhang, Jinliang Lu, Shaonan Wang, and Chengqing Zong. 2024. X-instruction: Aligning language model in low-resource languages with self-curated cross-lingual instructions. arXiv preprint arXiv:2405.19744.

Haonan Li, Fajri Koto, Minghao Wu, Alham Fikri Aji, and Timothy Baldwin. 2023b. Bactrian-x: A multilingual replicable instruction-following model with low-rank adaptation. arXiv preprint arXiv:2305.15011.

Petro Liashchynskyi and Pavlo Liashchynskyi. 2019. Grid search, random search, genetic algorithm: A big comparison for nas. arXiv preprint arXiv:1912.06059.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. 2023. The flan collection: Designing data and methods for effective instruction tuning. In International Conference on Machine Learning, pages 22631–22648. PMLR.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101.

Michael S Matena and Colin A Raffel. 2022. Merging models with fisher-weighted averaging. Advances in Neural Information Processing Systems, 35:17703–17716.

Sanket Vaibhav Mehta, Darshan Patil, Sarath Chandar, and Emma Strubell. 2021. An empirical investigation of the role of pre-training in lifelong learning. arXiv preprint arXiv:2112.09153.

Kurt Micallef, Nizar Habash, Claudia Borg, Fadhl Eryani, and Houda Bouamor. 2024. Cross-lingual transfer from related languages: Treating low-resource Maltese as multilingual code-switching. In Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1014–1025, St. Julian's, Malta. Association for Computational Linguistics.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2023. Crosslingual generalization through multitask finetuning. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 15991–16111, Toronto, Canada. Association for Computational Linguistics.

Thuat Nguyen, Chien Van Nguyen, Viet Dac Lai, Hieu Man, Nghia Trung Ngo, Franck Dernoncourt, Ryan A. Rossi, and Thien Huu Nguyen. 2024. CulturaX: A cleaned, enormous, and multilingual dataset for large language models in 167 languages. In Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), pages 4226–4237, Torino, Italia. ELRA and ICCL.

Xuan-Phi Nguyen, Wenxuan Zhang, Xin Li, Mahani Aljunied, Qingyu Tan, Liying Cheng, Guanzheng Chen, Yue Deng, Sen Yang, Chaoqun Liu, et al. 2023. Seallms–large language models for southeast asia. arXiv preprint arXiv:2312.00738.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia-Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

Marinela Parovic, Alan Ansell, Ivan Vulić, and Anna Korhonen. 2023. Cross-lingual transfer with target language-ready task adapters. In Findings of the Association for Computational Linguistics: ACL 2023, pages 176–193, Toronto, Canada. Association for Computational Linguistics.

Marinela Parović, Ivan Vulić, and Anna Korhonen. 2024. Investigating the potential of task arithmetic for cross-lingual transfer. In Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers), pages 124–137, St. Julian's, Malta. Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In Proceedings of the 2020 Conference on Empirical

Methods in Natural Language Processing (EMNLP), pages 7654–7673, Online. Association for Computational Linguistics.

Lütfi Kerem Senel, Benedikt Ebing, Konul Baghirova, Hinrich Schuetze, and Goran Glavaš. 2024. Kardeş-NLU: Transfer to low-resource languages with the help of a high-resource cousin – a benchmark and evaluation for Turkic languages. In Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1672–1688, St. Julian's, Malta. Association for Computational Linguistics.

Uri Shaham, Jonathan Herzig, Roee Aharoni, Idan Szpektor, Reut Tsarfaty, and Matan Eyal. 2024. Multilingual instruction tuning with just a pinch of multilinguality. arXiv preprint arXiv:2401.01854.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, et al. 2022. Language models are multilingual chain-of-thought reasoners. In The Eleventh International Conference on Learning Representations.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. arXiv preprint arXiv:1909.08053.

Shivalika Singh, Freddie Vargus, Daniel Dsouza, Börje F Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura OMahony, et al. 2024. Aya dataset: An open-access collection for multilingual instruction tuning. arXiv preprint arXiv:2402.06619.

Mingxu Tao, Yansong Feng, and Dongyan Zhao. 2023. Can BERT refrain from forgetting on sequential tasks? a probing study. In The Eleventh International Conference on Learning Representations.

Sebastian Thrun. 1998. Lifelong learning algorithms. In S. Thrun and L. Pratt, editors, Learning To Learn, pages 181 – 209. Kluwer Academic Publishers.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In International conference on machine learning, pages 23965–23998. PMLR.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. 2023. TIES-merging: Resolving interference when merging models. In Thirty-seventh Conference on Neural Information Processing Systems.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2024. Ties-merging: Resolving interference when merging models. Advances in Neural Information Processing Systems, 36.

Zheng Xin Yong, Hailey Schoelkopf, Niklas Muennighoff, Alham Fikri Aji, David Ifeoluwa Adelani, Khalid Almubarak, M Saiful Bari, Lintang Sutawika, Jungo Kasai, Ahmed Baruwa, Genta Winata, Stella Biderman, Edward Raff, Dragomir Radev, and Vassilina Nikoulina. 2023. BLOOM+1: Adding language support to BLOOM for zero-shot prompting. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 11682–11703, Toronto, Canada. Association for Computational Linguistics.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2023. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In Forty-first International Conference on Machine Learning.

Chen Zhang, Mingxu Tao, and Yansong Feng. 2024a. MLiC-Eval: An NLP Evaluation Suite for Minority Languages in China. https://github.com/luciusssss/MLiC-Eval.

Chen Zhang, Mingxu Tao, Quzhe Huang, Jiuheng Lin, Zhibin Chen, and Yansong Feng. 2024b. Mc^2: Towards transparent and culturally-aware nlp for minority languages in china. arXiv preprint arXiv:2311.08348.

Yiran Zhao, Wenxuan Zhang, Huiming Wang, Kenji Kawaguchi, and Lidong Bing. 2024. Adamergex: Cross-lingual transfer with large language models via adaptive adapter merging. arXiv preprint arXiv:2402.18913.

A Data Statistics

In Table A, we report the statistics of the evaluation datasets used in our study. We use the development set to tune the hyperparameters in the model merging algorithms and test the models on the test set.

We follow the license for the data used in our work. Our use of existing datasets is consistent with their intended use.

B Implementation Details

Since CulturaX and MC^2 are both cleaned and deduplicated corpus, we do not additionally preprocess these data. In this work, we employ Megatron-LM (Shoeybi et al., 2019) for continual pre-training and supervised fine-tuning. To obtain the LLMs adapted to each target language, i.e., the CT-X model mentioned in Section 3.2, we continually pre-train Llama-2-7B with the texts in corresponding language. Here we use AdamW (Loshchilov and Hutter, 2017) as the optimizer, with β_1 and β_2 are set to 0.9 and 0.95 respectively. The maximum learning rate is set to 2e-5, and the batch size is set to 1M tokens. We also use bfloat16 to train our models.

For model merging, we employ Arcee's MergeKit (Goddard et al., 2024) to merge the CT model in target language and the English SFT model. Following previous works (Yadav et al., 2023), we use grid search (Liashchynskyi and Liashchynskyi, 2019) to select the optimal hyperparameters. For the density of each LLM, i.e., k_1 and k_2 mentioned in Section 3.1, we select the hyperparameters from $\{0.01, 0.2, 0.4, 0.6, 0.8, 1.0\}$. For the scaling factor λ, Arcee's MergeKit can automatically normalize the magnitudes and self-adaptively selects the optimal scaling factor.

C Additional Experiment Results

C.1 Results of Individual Tasks

Here we report the performance of individual tasks for each language: Tamil in Table 7, Telugu in Table 8, Odia in Table 9, Bengali in Table 10, Tibetan in Table 11, Uyghur 12 and Mongolian in Table 13.

C.2 Case Study of Machine-Translated Data

Table 14 presents two examples of machine translation (MT) and human translation applied to SFT data. These examples illustrate the potential shortcomings of MT compared to human translation.

The MT output, generated by the NLLB model, often introduces irrelevant content and omits crucial information, as shown in the table's upper example. In the lower example, NLLB fails to translate several question options from English to Tibetan. These issues highlight the significant performance gap between human and machine translation.

Dataset	Dev	Test
MGSM	50	200
SIB-200	51	204
Belebele	720	2,880
TC in MLiC-Eval	48	504
MRC in MLiC-Eval	20	200
RS in MLiC-Eval	40	407
Math in MLiC-Eval	20	200

Table 6: Number of instances for each language in the evaluation datasets. TC is short for text classification. MRC is short for machine reading comprehension. RS is short for response selection.

	SIB-200	Belebele	Average
BASE	30.88	25.42	28.15
SFT-flan	30.88	27.50	29.19
CT-tam	73.53	30.83	52.18
CTSFT-tam-mt	69.61	31.53	50.57
CTSFT-tam-flan	71.08	36.81	53.95
WAVG-tam-flan	80.39	34.72	57.56
TIES-tam-flan	80.39	36.53	58.46

Table 7: The performance of different models on the Tamil tasks.

	MGSM	SIB-200	Belebele	Average
BASE	2.50	27.45	26.53	18.83
SFT-flan	1.00	24.02	26.81	17.28
CT-tel	2.00	73.53	28.47	34.67
CTSFT-tel-mt	3.00	61.27	34.44	32.90
CTSFT-tel-flan	7.00	77.45	29.44	37.96
WAVG-tel-flan	3.00	76.96	32.78	37.58
TIES-tel-flan	8.00	77.45	33.06	39.50

Table 8: The performance of different models on the Telugu tasks.

	SIB-200	Belebele	Average
BASE	26.47	26.81	26.64
SFT-flan	25.98	24.44	25.21
CT-ory	69.61	26.25	47.93
CTSFT-ory-mt	32.25	27.92	30.14
CTSFT-ory-flan	61.76	27.36	44.56
WAVG-ory-flan	77.45	29.72	53.59
TIES-ory-flan	81.86	31.11	56.49

Table 9: The performance of different models on the Odia tasks.

	MGSM	SIB-200	Belebele	Average
BASE	1.00	50.00	25.42	25.47
SFT-flan	1.50	47.06	25.97	24.84
CT-ben	1.50	63.73	27.08	30.77
CTSFT-ben-mt	7.00	76.96	31.25	38.40
CTSFT-ben-flan	7.50	80.88	38.19	42.19
WAVG-ben-flan	3.50	76.96	31.11	37.19
TIES-ben-flan	7.50	79.41	34.02	40.31

Table 10: The performance of different models on the Bengali tasks.

	TC	MRC	RS	Math	Average
BASE	0.60	28.50	22.36	2.50	13.49
SFT-flan	14.48	44.00	32.68	2.00	23.29
CT-bod	0.40	28.00	18.18	7.50	13.52
CTSFT-bod-mt	48.41	42.50	30.47	14.00	33.85
CTSFT-bod-flan	70.24	46.50	45.70	7.00	42.36
WAVG-bod-flan	74.40	51.50	40.79	10.50	44.30
TIES-bod-flan	78.17	56.00	42.26	15.00	47.86

Table 11: The performance of different models on the Tibetan tasks. All four tasks are from MLiC-Eval. TC is short for text classification. MRC is short for machine reading comprehension. RS is short for response selection.

	TC	MRC	RS	Math	Average
BASE	0.00	26.00	23.34	4.00	13.34
SFT-flan	7.94	43.00	34.15	4.00	22.27
CT-uig	1.39	21.50	25.31	11.00	14.80
CTSFT-uig-mt	20.63	37.00	28.26	13.50	24.85
CTSFT-uig-flan	90.28	53.50	38.57	15.50	49.46
WAVG-uig-flan	54.56	54.00	42.01	20.00	42.64
TIES-uig-flan	87.50	56.00	45.21	21.00	52.43

Table 12: The performance of different models on the Uyghur tasks. All four tasks are from MLiC-Eval. TC is short for text classification. MRC is short for machine reading comprehension. RS is short for response selection.

	TC	MRC	RS	Math	Average
BASE	0.40	21.50	21.38	3.00	11.57
SFT-flan	10.52	34.50	33.42	1.00	19.86
CT-mvf	0.60	12.5	27.76	3.50	11.09
CTSFT-mvf-mt	20.24	31.00	27.03	0.00	19.57
CTSFT-mvf-flan	40.87	30.00	25.80	0.50	24.29
WAVG-mvf-flan	40.48	45.00	35.87	3.00	30.09
TIES-mvf-flan	45.04	50.00	32.18	3.00	32.56

Table 13: The performance of different models on the Mongolian tasks. All four tasks are from MLiC-Eval. TC is short for text classification. MRC is short for machine reading comprehension. RS is short for response selection.

English
***Input*:** Give the rationale and then the answer. Is the following statement true?
"The United States Army Air Forces sent aircraft into the skies."
***Output*:** The United States Army Air Forces were responsible for aerial military. The final answer: yes.

Uyghur (Translated by the NLLB Model)
***Input*:**

(ئى مۇھەممەد!) بۇ سۆزنىڭ (راستلىقى) ئىسپاتلىنامدۇ؟ ئاندىن ئۇنىڭ (راستلىقى) غا جاۋاب قايتۇرغىن.

***Output*:**

ئامېرىكا ئارمىيەسى ھاۋا ئارمىيەسى ھاۋا ئارمىيەسىنىڭ ھاۋا ئارمىيەسى بىلەن مەسئۇل ئىدى

Uyghur (Translated by Native Speakers)
***Input*:**

سەۋەبىنى كۆرسىتىپ ئاندىن جاۋاب بېرىڭ. تۆۋەندىكى بايانلار توغرىمۇ؟
" ئامېرىكا قۇرۇقلۇق ئارمىيە ئاۋىئاتسىيە ئەترىتى ئايرۇپىلاننى بوشلۇققا چىقاردى."

***Output*:**

ئامېرىكا قۇرۇقلۇق ئارمىيە ئاۋىئاتسىيە ئەترىتى بوشلۇقتىكى ھەربىي ھەرىكەتكە مەسئۇل.
ئەڭ ئاخىرقى جاۋابى: توغرا.

English
***Input*:** What do people usually tie a ribbon around as a present?
A. floral arrangement B. girl's hair C. box
D. car E. typewriter
The best answer is
Answer:
***Output*:** A

Tibetan (Translated by the NLLB Model)
***Input*:**

མི་ཚོས་རས་ཀྱི་སྐུད་པར་ཇི་ཞིག་བཏགས་པ་ཡིན་ནམ།
A. ལེ་ཏོག་གི་སྒྲོམ་ B. བུ་མོ་གི་སྐུ་ C. Box D. Car E. Typewriter
ལན་གསལ་ལེགས་པོ་འདི་ཡིན།

***Output*:** A

Tibetan (Translated by Native Speakers)
***Input*:**

མི་རྣམས་ཀྱིས་རྒྱན་པར་དར་སྐུད་ཅ་ལག་གང་ཞིག་གི་སྟེང་དུ་བསྡམས་ནས་ལག་རྟགས་བྱེད་བཞིན་ཡོད་དམ།
A.ལེ་ཏོག B.བུ་མོའི་སྐུ། C.སྒམ།
D.རླངས་འཁོར། E.ཡིག་གཉགས་འཕྲལ་འཁོར།
དེས་ལན་ཡག་པོགོ་ཞེ!
ལན།

***Output*:** A

Table 14: Translation samples of Uyghur (upper) and Tibetan (lower) SFT data.