# LayAlign: Enhancing Multilingual Reasoning in Large Language Models via Layer-Wise Adaptive Fusion and Alignment Strategy

**Zhiwen Ruan**[1*], **Yixia Li**[1*], **He Zhu**[2], **Longyue Wang**[3], **Weihua Luo**[3]
**Kaifu Zhang**[3], **Yun Chen**[4], **Guanhua Chen**[1†]
[1]Southern University of Science and Technology, [2]Peking University
[3]Alibaba International Digital Commerce,[4]Shanghai University of Finance and Ecnonmics

## Abstract

Despite being pretrained on multilingual corpora, large language models (LLMs) exhibit suboptimal performance on low-resource languages. Recent approaches have leveraged multilingual encoders alongside LLMs by introducing trainable parameters connecting the two models. However, these methods typically focus on the encoder's output, overlooking valuable information from other layers. We propose Layer-Wise Adaptive Fusion and Alignment Strategy (LayAlign), a framework that integrates representations from all encoder layers, coupled with the adaptive fusion-enhanced attention mechanism to enable layer-wise interaction between the LLM and the multilingual encoder. Extensive experiments on multilingual reasoning tasks, along with analyses of learned representations, show that our approach consistently outperforms existing baselines.

## 1 Introduction

Large Language Models (LLMs) are predominantly trained on corpora emphasizing a select group of high-resource languages, enabling them to demonstrate strong reasoning capabilities in tasks such as mathematics (Yu et al., 2024; Azerbayev et al., 2024; Luo et al., 2023; Mitra et al., 2024) and commonsense reasoning (Finch and Choi, 2024; Fang et al., 2024). However, most of these LLMs are derived from English-centric models and fine-tuned using English-specific downstream data. Consequently, their performance in low-resource languages remains significantly limited, leading to pronounced disparities between high-resource and low-resource language capabilities.

While multilingual pretrained models attempt to bridge this gap by supporting a broader set of languages, they often exhibit limited reasoning abilities due to constrained training data and model parameters (Xue et al., 2021; Team et al., 2022). In contrast, English-centric LLMs display strong reasoning skills but struggle with multilingual understanding, leading to poorer performance in low-resource languages. Inspired by multimodal approaches (Alayrac et al., 2022; Liu et al., 2023; Chen et al., 2024a; Zhou et al., 2024), works like LangBridge (Yoon et al., 2024) and MindMerger (Huang et al., 2024) aim to enhance multilingual reasoning by integrating a multilingual encoder (Xue et al., 2021) with an LLM via a trainable adapter. However, these methods focus only on the top multilingual encoder layer, overlooking the potential richness of intermediate representations.

In this paper, we introduce Layer-Wise Adaptive Fusion and Alignment Strategy (LayAlign), a framework that integrates representations from all multilingual encoder layers by applying distinct fusion ratios for each LLM layer. This approach enables the model to leverage both low- and high-level representations effectively. To incorporate the fused multilingual representations into the decoder-only LLM, we propose a adaptive fusion-enhanced attention mechanism combining cross-attention and self-attention. This mechanism uses representations from the layer-wise aligner to generate key-value pairs, with learnable gate parameters modulating cross-attention intensity.

LayAlign is optimized with a two-stage finetuning scheme, keeping both the multilingual encoder and LLM backbone frozen. LayAlign encourages the model to select representations from appropriate encoder layers, facilitating a shared multilingual representation space across all LLM layers. We evaluate the effectiveness of LayAlign on mathematical and commonsense reasoning tasks. Our experimental results and analyses of the learned representation space demonstrate that LayAlign significantly improves reasoning performance for low-resource languages while maintaining strong

[*]Equal Contribution.
[†]Corresponding Author.

*Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1481–1495
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics

results for high-resource languages.[1]

## 2 Related Work

### 2.1 Multilingual Large Language Models

To address the demand for supporting global linguistic diversity, researchers have expanded into multilingual LLMs (Qin et al., 2025). Advanced models like Qwen2 (Yang et al., 2024) and LLaMA3 (AI@Meta, 2024) support multiple languages, showcasing robust multilingual capabilities. However, these models are trained from scratch, which incurs substantial computational costs and requires extensive datasets for relevant languages, often leading to inadequate support for low-resource languages. These meticulously trained models frequently face challenges in scaling to other languages, particularly those with lower representation in the training data.

Recently, LangBridge (Yoon et al., 2024) and MindMerger (Huang et al., 2024) feature an English-centric LLM backbone, a multilingual encoder that offers multilingual information, and an adapter that facilitates interoperability between the multilingual and English languages. However, these approaches are limited to representations from the topmost encoder layer, neglecting potentially valuable insights from other layers. Our LayAlign framework follows this line and explores to better leverage the multilingual information of different encoder layers to enhance the multilingual reasoning abilities of LLMs.

### 2.2 Aligning Pretrained Representations

The integration of encoders with large language models (LLMs) has been widely studied in the cross-modal domain (Liu et al., 2023; Chen et al., 2024a; Zhou et al., 2024). Many approaches utilize vision-to-language adapter modules to align visual and textual modalities, mapping the output of vision encoders to the soft prompt inputs of LLMs. Other works employ cross-attention mechanisms to enable more direct interaction between image and text representations (Alayrac et al., 2022; Chen et al., 2024b). Drawing inspiration from these cross-modal strategies, our method enhances multilingual reasoning by integrating a multilingual encoder with an LLM. To bridge the gap between these components, we introduce an aligner that enables efficient interaction via cross-attention.

## 3 Method

We introduce LayAlign, which facilitates the direct interaction between all layers of the LLM and the representations of the multilingual encoder through a layer-wise aligner and adaptive fusion-enhanced attention. This approach allows for a more comprehensive integration of language comprehension information from the encoder, thereby enhancing the multilingual reasoning capabilities of LLM. In the subsequent sections, we provide a detailed overview of our framework, focusing on the model architecture (Section 3.1), adaptive fusion-enhanced attention (Section 3.2), and training methodology (Section 3.3).

### 3.1 Model Architecture

As depicted in Figure 1, the adapter and layer-wise aligner are designed to align a multilingual encoder with $n$ layers to the representation space of an LLM with $m$ layers. The input multilingual text $I_{\text{in}}$ is processed by the encoder, producing a series of representations $\{H_1, H_2, \ldots, H_n\}$, where $H_i$ denotes the output of the $i$-th encoder layer. Following prior work (Yoon et al., 2024), an adapter is employed to map the final layer's representation $H_n$ to the soft prompt input $I_{\text{map}}$ for the LLM, thereby enhancing multilingual reasoning capabilities, where $I_{\text{map}} = \text{Adapter}(H_n)$.

However, this approach only utilizes the final layer representation $H_n$, disregarding the intermediate representations from the embedding $H_0$ through the encoder layers $H_1, \ldots, H_{n-1}$. To fully harness the multilingual potential of the encoder, we propose a novel layer-wise aligner that explicitly integrates both low-level and high-level representations from multiple layers of the multilingual encoder, rather than relying solely on the final layer's output.

For each LLM layer, the layer-wise aligner generates a fused representation by assigning distinct weights to different multilingual encoder layers. This mechanism allows the model to learn the optimal combination of low-level and high-level features across encoder layers, establishing a correspondence between the multilingual encoder and each LLM layer. While the adapter leverages the final layer representation of the encoder, the layer-wise aligner integrates information from the embedding layer and intermediate encoder layers, enriching the LLM with additional multilingual context.

[1] Our code and models are publicly available at https://github.com/sustech-nlp/LayAlign.

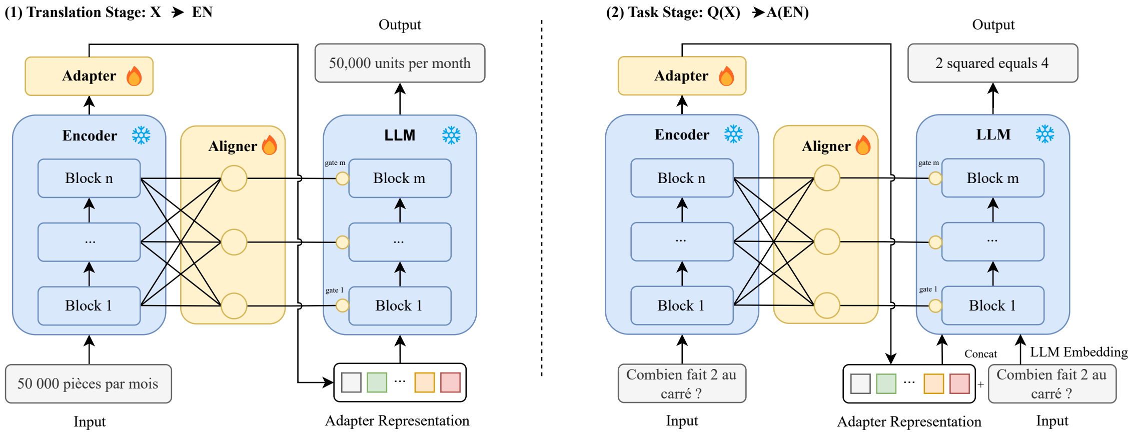


Figure 1: Overview of LayAlign. A multilingual encoder is aligned with the target LLM with an adapter and the layer-wise aligner. We keep the multilingual encoder and LLM frozen, whereas the adapter and layer-wise aligner are optimized in two stages. For simplicity, shifted output tokens were omitted from the input representation. **Left:** (1) Translation stage. In this stage, LayAlign is fine-tuned using translation data, where the data consists of translations from other languages into English. **Right:** (2) Task Stage. In this stage, LayAlign is fine-tuned using specialized downstream task data, where the input is multilingual and the output is in English.

Formally, the fusion process is defined as:

$$H_i^K, H_i^V = f_i(H_0, ..., H_{n-1}), \qquad (1)$$

where $f_i(\cdot)$ is the fusion function for $i$-th layer of LLM, responsible for fusing $\{H_0, ..., H_{n-1}\}$ into the fused representations $\{H_i^K, H_i^V\}$. Specifically, $f_i(\cdot)$ consists of two linear layers with a ReLU activation in between. The resulting $H_i^K$ and $H_i^V$ are then fed into the $i$-th layer of the LLM as keys and values for cross-attention computing. The detailed procedure is provided in Section 3.2.

## 3.2 Adaptive Fusion-Enhanced Attention

We denote the hidden states in the $i$-th LLM decoder layer as $T_i$, where $i \in [0, m]$ and $T_0$ denotes the concatenation of the output from the decoder's embedding layer with the output from the adapter module. The final representation $T_m$ is utilized to generate the next token. In these transformer layers, standard self-attention (SA) is employed. However, it can not directly interact with the fused representations from the multilingual encoder. To address this limitation, we replace the vanilla attention mechanism in all transformer layers with adaptive fusion-enhanced attention, which incorporates self-attention, cross-attention, and a gate module, as shown in Figure 2.

Specially, for the $i$-th LLM layer representation, the attention mechanism is computed as follows:

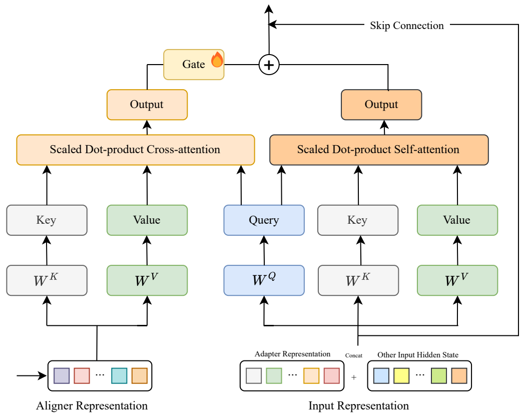


Figure 2: The illustration of our proposed Adaptive Fusion-Enhanced Attention. It consists of self-attention (right), cross-attention (left), and a gate module. Both cross-attention and self-attention modules share the same linear weights as that of the backbone LLM.

$$\begin{aligned} GA(T_{i-1}, H_i^K, H_i^V) &= SA(T_{i-1}) \\ &+ g_i \cdot CA(T_{i-1}, H_i^K, H_i^V) \end{aligned} \qquad (2)$$

Here, keys $H_i^K$ and values $H_i^V$ represent the fused multilingual encoder representations generated by the layer-wise aligner for the $i$-th layer of the LLM. $T_{i-1}$ is the output of the $i-1$-th layer of the LLM, while SA and CA denote self-attention and cross-attention, respectively. A learnable gate, $g_i$, is introduced to regulate the incorporation of fused information into the $i$-th layer automatically. This gate is initialized to 0 to ensure smooth training at the initial stages. The cross-attention module shares the same linear parameters $W^Q$, $W^K$, $W^V$, and $W^O$ as the self-attention module, thus eliminating the need for additional parameters. These parameters remain frozen during training. For clarity, the output linear matrix $W^O$ is omitted in Figure 2.

### 3.3 Two-stage Training

The training process is structured into two distinct phases, as illustrated in Figure 1. The first phase, referred to as the **translation stage**, concentrates on aligning the representation spaces between the multilingual encoder and the LLM. During this stage, LayAlign is fine-tuned using parallel corpora from the many-to-English machine translation task. The input to the LLM is derived from the adapter's output $I_{\text{map}}$, denoted as $X = [\langle bos \rangle; I_{\text{map}}; \langle sep \rangle]$.

The second phase termed the **task stage**, is designed to enhance the model's performance on specific downstream tasks within a multilingual context. In this phase, LayAlign is fine-tuned on specialized downstream task data, where the input is multilingual and the output is in English. Here, the LLM's input combines both the adapter's output and the original user input text $I_{\text{in}}$ (as shown in Figure 1), represented as $X = [\langle bos \rangle; I_{\text{map}}; \langle sep \rangle; \text{embed}(I_{\text{in}})]$. Unlike baseline approaches such as LangBridge (Yoon et al., 2024), which rely solely on the adapter's output $I_{\text{map}}$ as the LLM input, this approach incorporates additional context, fostering task-specific adaptation and improved multilingual performance. The impact of the additional LLM input $I_{\text{in}}$ is further examined in Section 5.2.

## 4 Experiments

We compare LayAlign with baselines on mathematical reasoning, commonsense reasoning, and language understanding tasks following prior studies (Huang et al., 2024; Yoon et al., 2024).

### 4.1 Mathematical Reasoning

#### 4.1.1 Experimental Setup

**Evaluation Dataset.** In line with Huang et al. (2024), we employ two datasets for evaluating LayAlign: MGSM (Shi et al., 2023) and MSVAMP (Chen et al., 2023a). MGSM contains multilingual grade school-level mathematical word problems, while MSVAMP serves as an out-of-domain evaluation set, providing a broader assessment of the multilingual mathematical reasoning capabilities. Models are evaluated using a zero-shot approach.

| Method | Backbone | Training Data | Source |
|---|---|---|---|
| **English-Only Data Baselines** | | | |
| MetaMath | LLaMA2-7B | MetaMathQA | Official checkpoint[2] |
| LangBridge-EN | `mT5-xl`+MetaMath | MetaMath-200k | Yoon et al. (2024) |
| Translate | NLLB+MetaMath | None | Reimplementation |
| **Multi-lingual Data Baselines** | | | |
| MetaMath-Mul | MetaMath | MultilingualMath | Reimplementation |
| MathOctopus | LLaMA2-7B | MGSM8KInstruct | Official checkpoint[3] |
| LangBridge | `mT5-xl`+MetaMath | Lego-MT+MultilingualMath | Reimplementation |
| MindMerge | `mT5-xl`+MetaMath | Lego-MT+MultilingualMath | Official checkpoint[4] |

Table 1: Comparisions of baselines. LangBridge and MindMerge are trained with the same two-stage data as LayAlign.

**Training Datasets.** Consistent with the setup in MindMerger (Huang et al., 2024), we leverage the same training data for LayAlign. In the first stage, the model is trained on translation data from the Lego-MT corpus (Yuan et al., 2023), which translates multilingual inputs into English. In the second stage, we employ the composite multilingual mathematical data, referred to as MultilingualMath (Yu et al., 2024; Chen et al., 2023a), consisting of 30,000 samples per language across ten languages. This dataset supports comprehensive training for robust multilingual mathematical reasoning.

**Baselines.** We compare our approach against seven baselines.

- **MetaMath**: **MetaMath** is fine-tuned from LLaMA2-7B on MetaMathQA, a mathematical dataset derived from GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021). We further train MetaMath on our second phase multilingual task data, resulting in **MetaMath-Mul**.
- **Translate** (Shi et al., 2023): a training-free method that translates the prompt into English for MetaMath. We utilize NLLB-200-3.3B (Team et al., 2022) as the translator followed MindMerger.
- **MathOctopus** (Chen et al., 2023b): fine-tuned from LLaMA2-7B on a custom multilingual mathematical reasoning dataset. We utilize their best-performing checkpoint xRFT-MathOctopus.
- **LangBridge** (Yoon et al., 2024): aligns `mT5-xl` with MetaMath by projecting the final-layer hidden states of `mT5-xl` into MetaMath's input via an adapter. We compare against both **LangBridge-EN**, the original model trained on the English dataset MetaMath-200k, and **LangBridge**, which we trained on the same datasets as LayAlign using

[2] https://huggingface.co/meta-math/MetaMath-7B-V1.0

[3] https://huggingface.co/Mathoctopus/Parallel_xRFT_7B

[4] https://github.com/CONE-MT/MindMerger

our two-stage training process for a fair comparison.

- **MindMerger** (Huang et al., 2024): it shares a similar architecture with LangBridge. While LangBridge feeds multilingual math prompts exclusively into `mT5-xl`, MindMerger processes the prompts in parallel through both `mT5-xl` and MetaMath at the second stage.

**Model and Training Details.** We utilize the encoder of `mT5-xl` as the multilingual encoder, comprising 1.6 billion parameters, with MetaMath (Yu et al., 2024) serving as the LLM. The training procedure is conducted in two stages. During the first stage, the learning rate is set to $4 \times 10^{-5}$, with a batch size of 128, over 3 epochs, and a warmup ratio of 0.05. In the second stage, the learning rate is adjusted to $3 \times 10^{-5}$, while maintaining the batch size at 128, the number of epochs at 3, and the warmup ratio at 0.05. All experiments are executed on 8 NVIDIA L40 GPUs, with the first and second stages taking 9 and 8 hours, respectively.

### 4.1.2 Results

Table 2 presents the results of the mathematical reasoning tasks. As shown, LayAlign significantly surpasses all baselines, outperforming the current state-of-the-art, MindMerger, by 1.6% on MGSM and 1.1% on MSVAMP in terms of average accuracy across all languages. These results highlight the effectiveness of LayAlign on English LLM.

For methods that directly finetune LLMs, such as MetaMath, MetaMath-Mul, and MathOctopus, it is challenging to achieve strong performance across both high-resource and low-resource languages simultaneously. Training exclusively in English (e.g., MetaMath) generally results in high performance for high-resource languages like English, but poor results in low-resource languages. Conversely, methods trained on multilingual data (e.g., MetaMath-Mul and MathOctopus) often suffer from a significant performance drop in high-resource languages. For instance, MetaMath-Mul's performance declines by 11.8 and 21.6 points on high-resource languages in the MGSM and MSVAMP datasets, respectively. This demonstrates the difficulty of achieving consistently high performance across both high-resource and low-resource languages in LLM-based models.

This challenge can be significantly alleviated by models such as LangBridge, MindMerger, and LayAlign, which share a common architecture that integrates a multilingual encoder with LLMs. All three models demonstrate substantial improvements over traditional LLM-based approaches in both high- and low-resource languages. Among them, LayAlign achieves the best performance on the MGSM and MSVAMP benchmarks, highlighting its ability to effectively leverage the representations from the multilingual encoder through the layer-wise aligner and adaptive fusion-enhanced attention mechanisms.

We further compare LayAlign with Translate. LayAlign surpasses Translate by a substantial margin, showing improvements of 15.9 points on MGSM and 10.1 points on MSVAMP. Additional, Translate suffers from longer inference times and reliance on external translation systems due to its need to translate multilingual prompts into English.

## 4.2 Commonsense Reasoning and Language Understanding

### 4.2.1 Experimental Setup

**Evaluation Datasets.** We evaluate commonsense reasoning and language understanding capabilities using X-CSQA (Lin et al., 2021) and XNLI (Conneau et al., 2018), respectively.

**Training Datasets.** For both tasks, we adopt the same dataset setup as MindMerger (Huang et al., 2024). The Lego-MT translation dataset (Yuan et al., 2023) is utilized in the first training stage, while the translated X-CSQA training set (Huang et al., 2022, 2024) and the official development set of XNLI are used in the second training stage for the commonsense reasoning and language understanding tasks, respectively.

**Baselines.** We compare LayAlign with two LLM-based baselines:

- **LLaMAX2** (Lu et al., 2024): fine-tuned from the powerfull multilingual model LLaMAX2-7B on an English task dataset, with LLaMAX2-7B covering all the languages examined in this study. We utilize the official checkpoints.[5].
- **LLaMAX2-Mul**: fine-tuned from LLaMAX2 using the same multilingual task dataset as ours.

We also include LangBridge and MindMerger in our comparisons. To ensure a fair evaluation, all models, including LangBridge, MindMerger, and LayAlign, utilize LLaMAX2 as the LLM and

[5] Commonsense reasoning: https://huggingface.co/LLaMAX/LLaMAX2-7B-X-CSQA; Language understanding: https://huggingface.co/LLaMAX/LLaMAX2-7B-XNLI

| **MGSM** | **Avg.** | **Lrl.** | **Hrl.** | **Bn** | **Th** | **Sw** | **Ja** | **Zh** | **De** | **Fr** | **Ru** | **Es** | **En** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MetaMath | 37.9 | 5.9 | 51.6 | 6.4 | 6.4 | 4.8 | 34.8 | 39.2 | 56.4 | 55.6 | 51.6 | 55.2 | **68.4** |
| LangBridge-EN | 50.2 | 45.5 | 52.3 | 42.8 | 50.4 | 43.2 | 40.0 | 45.2 | 50.8 | 52.4 | 56.4 | 58.0 | 63.2 |
| Translate | 43.1 | 36.1 | 46.1 | 46.4 | 27.2 | 34.8 | 28.4 | 34.8 | 48.8 | 44.0 | 42.4 | 55.6 | 68.4 |
| MetaMath-Mul | 38.4 | 35.1 | 39.8 | 32.0 | 36.8 | 36.4 | 35.2 | 40.0 | 40.8 | 41.2 | 39.6 | 40.8 | 41.2 |
| MathOctopus | 40.0 | 33.5 | 42.8 | 30.4 | 35.2 | 34.8 | 38.0 | 45.6 | 41.6 | 38.4 | 39.6 | 46.0 | 50.4 |
| LangBridge | 54.0 | 50.1 | 55.6 | 48.8 | 49.2 | 52.4 | 50.0 | 53.6 | 56.0 | 54.0 | 58.0 | 58.0 | 59.6 |
| MindMerger | 57.4 | 54.8 | 58.6 | 51.2 | 56.8 | 56.4 | 50.8 | 54.4 | 60.0 | 55.2 | **62.4** | 59.6 | 67.6 |
| **LayAlign** | **59.0** | **56.4** | **60.2** | **51.6** | **59.2** | **58.4** | **52.0** | **56.0** | **62.0** | **61.6** | 61.6 | **61.6** | 66.4 |

| **MSVAMP** | **Avg.** | **Lrl.** | **Hrl.** | **Bn** | **Th** | **Sw** | **Ja** | **Zh** | **De** | **Fr** | **Ru** | **Es** | **En** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MetaMath | 47.5 | 15.5 | **61.2** | 13.5 | 16.1 | 16.9 | 53.9 | 56.2 | **63.7** | **64.9** | 57.8 | **64.6** | **67.6** |
| LangBridge-En | 52.0 | 45.1 | 54.9 | 46.8 | 46.3 | 42.1 | 45.5 | 50.4 | 58.1 | 57.0 | 55.8 | 56.9 | 60.6 |
| Translate | 49.0 | 44.5 | 50.9 | 49.3 | 44.2 | 40.1 | 42.0 | 48.0 | 46.5 | 49.5 | 45.1 | 57.9 | 67.6 |
| MetaMath-Mul | 37.8 | 33.7 | 39.6 | 30.0 | 35.1 | 36.0 | 38.3 | 37.9 | 39.1 | 41.4 | 39.0 | 41.7 | 39.8 |
| MathOctopus | 38.1 | 33.1 | 40.3 | 27.3 | 32.9 | 39.1 | 39.2 | 38.2 | 40.1 | 43.2 | 38.8 | 41.4 | 41.1 |
| LangBridge | 54.4 | 51.6 | 55.5 | 49.9 | 52.2 | 52.7 | 53.3 | 54.1 | 56.0 | 56.4 | 54.7 | 56.1 | 58.1 |
| MindMerger | 58.0 | 53.1 | 60.2 | **52.2** | 53.2 | 53.9 | 57.3 | 57.0 | 61.3 | 60.2 | 58.1 | 62.9 | 64.3 |
| **LayAlign** | **59.1** | **54.6** | 61.1 | 51.8 | **55.1** | **56.9** | **59.3** | **58.7** | 62.5 | 62.1 | **58.8** | 62.0 | 64.0 |

Table 2: Experimental results on MGSM and MSVAMP datasets. 'Lrl.', 'Hrl.', and 'Avg.' represent the average accuracy across low-resource languages, high-resource languages, and all languages, respectively. Referring to Huang et al. (2024), we regard Bn, Th, and Sw as low-resource languages, and regard the remaining languages as high-resource languages. Models above the line are trained in English, while those below are trained in multiple languages. The languages corresponding to the abbreviations used in the tables are provided in Appendix B.

| X-CSQA | Avg. | Sw | Fr | En |
|---|---|---|---|---|
| LLaMAX2 | 55.0 | 43.1 | 61.4 | 73.9 |
| LLaMAX2-Mul | 49.4 | 39.2 | 53.4 | 68.6 |
| LangBridge | 56.7 | 52.5 | 60.4 | 62.4 |
| MindMerger | 61.2 | 51.5 | 64.5 | 75.6 |
| LayAlign | **62.3** | **53.3** | **66.5** | **76.7** |

| XNLI | Avg. | Sw | Fr | En |
|---|---|---|---|---|
| LLaMAX2 | 76.5 | 66.7 | 83.1 | **89.7** |
| LLaMAX2-Mul | 77.4 | 68.3 | **84.7** | 89.3 |
| LangBridge | 76.0 | 72.2 | 78.0 | 80.8 |
| MindMerger | 79.2 | 72.7 | 84.2 | 88.5 |
| LayAlign | **79.7** | **73.0** | **84.7** | 88.9 |

Table 3: Experimental results on X-CSQA and XNLI datasets. Due to limited space, we list several representative languages in this table. The complete results is in Table 12 and Table 13 of Appendix B.

`mT5-xl` encoder as the multilingual encoder, with all models trained on the same two-stage dataset.

### 4.2.2 Results

The results for X-CSQA and XNLI are presented in Table 3. As shown, LayAlign sets a new state-of-the-art, outperforming LangBridge by 9.9% and MindMerger by 1.8% on X-CSQA, and improving by 4.9% and 0.6%, respectively, on XNLI. These results demonstrate that LayAlign is effective not only on English LLM backbones but also on multilingual LLM backbones.

Since the LLM backbone is inherently multilingual across all languages tested, fine-tuning it on English task datasets already yields strong multilingual task performance. For example, LLaMAX2 achieves scores of 55.0 on X-CSQA and 76.5 on XNLI. This makes further improvements challenging, as both LLaMAX2-Mul, which is fine-tuned on the multilingual task data, and LangBridge, which integrates a multilingual encoder into LLaMAX2, show only marginal gains or even performance declines. In contrast, LayAlign delivers robust performance in both commonsense reasoning and language understanding, underscoring the effectiveness of the layer-wise aligner, adaptive fusion-enhanced attention, and LLM text input integration.

| MGSM | Avg. | Lrl. | Hrl. |
|---|---|---|---|
| w/o Adapter | 44.1 | 15.9 | 56.2 |
| w/o LLM Input | 56.8 | 55.5 | 57.3 |
| w/o Layer-Wise Aligner | 56.9 | 53.1 | 58.6 |
| w/o Translation Stage | 52.0 | 38.9 | 57.5 |
| w/o Task Stage | 38.8 | 24.7 | 44.9 |
| MetaMath | 37.9 | 5.9 | 51.6 |
| LayAlign | 59.0 | 56.4 | 60.2 |

Table 4: Ablation experiments of LayAlign on the MGSM dataset. The complete table of accuracy for each language is in Table 15.

| MGSM | Parm(M) | Avg. | Lrl. | Hrl. |
|---|---|---|---|---|
| mGPT | 1418 | 48.5 | 30.8 | 56.1 |
| XGLM | 1733 | 51.1 | 42.4 | 54.8 |
| NLLB | 1733 | 55.3 | 50.8 | 57.2 |
| mT5-xl | 1670 | 59.0 | 56.4 | 60.2 |

Table 5: Experiments on MGSM using MetaMath as LLM and different multilingual models as encoders. The complete table for each language is in Table 14.

### 4.3 Ablation Studies

We conduct ablation studies to examine the contributions of key components in our method, including the adapter, the layer-wise aligner, the LLM input embedding $I_{\text{in}}$, and the two-stage training approach. Table 4 presents the ablation results. Note that the layer-wise aligner and adaptive fusion-enhanced attention operate in conjunction, so removing the layer-wise aligner also disables adaptive fusion-enhanced attention. As shown in Table 4, all components significantly contribute to LayAlign's overall performance. Since the adapter receives the highest-level representations from the encoder, its removal results in a substantial performance drop of 7.0 points. The task fine-tuning stage, which is directly related to downstream evaluations, also plays a critical role in the model's success. The layer-wise aligner and the translation stage are all integral to LayAlign, with their absence leading to performance declines of 2.1, 2.2, and 7.0 points, respectively. Notably, even without task-specific fine-tuning, LayAlign outperforms MetaMath by 33.0 points on low-resource languages, demonstrating that aligning the multilingual encoder with the LLM enhances task performance in low-resource settings, even in the absence of specialized training. To evaluate the role of the gating mechanism in LayAlign, we also conduct an ablation study by removing the gate. Without the gate, we observe that the training loss of LayAlign fails to decrease effectively. This highlights the gating mechanism's critical role in ensuring smooth and stable training.

## 5 Analyses

### 5.1 Multilingual Encoder

The LayAlign framework allows for the flexible selection of various multilingual models as encoders to extract multilingual representations. We evaluated several multilingual models on the MGSM benchmark, including the encoder from the two encoder-decoder multilingual models `mT5-xl` (Xue et al., 2021) and `NLLB-200-3.3B` (Team et al., 2022), as well as the decoder-only architectures mGPT (Shliazhko et al., 2023) and XGLM (Lin et al., 2022). As shown in Table 5, the encoder from `mT5-xl` achieves the best performance, while the encoders from the encoder-decoder multilingual models generally outperform those using multilingual decoders as LayAlign's encoder.

### 5.2 Training on English Task Data

In prior experiments, we demonstrated the effectiveness of LayAlign under multilingual training conditions. However, obtaining task-specific data for low-resource languages remains a significant challenge. To address this, we examine the performance of LayAlign when trained exclusively on English task-specific data by replacing the task-stage training set with the English MetaMath-200k dataset. Since both the input and output are in English, the LLM input could act as a shortcut for the model, potentially harming the learning of the multilingual aligner and adapter during finetuning. This may lead to poor performance in low-resource languages. Conversely, removing the LLM input text forces the model to depend on the multilingual encoder, encouraging cross-lingual generalization. To verify this, we evaluate three variants of LayAlign on the MGSM benchmark: the full LayAlign model, LayAlign without the LLM input text $I_{\text{in}}$, and LangBridge, which serves as a baseline equivalent to LayAlign without both the LLM input and the layer-wise aligner.

As shown in Table 6, when trained in an English-only setting, LayAlign tends to exploit the shortcut by relying heavily on the English LLM input. As a result, the multilingual information from the encoder is largely ignored during finetuning, leading to poor performance on low-resource languages. In contrast, the LayAlign variant without LLM input text is forced to rely on the multilingual information provided by the `mT5` encoder during finetuning. The superior performance of this variant underscores the critical importance of the layer-wise aligner, particularly in English-only downstream finetuning. In this setting, the LayAlign variant without LLM input text is recommended to enhance the model's multilingual capabilities, as it effectively leverages the multilingual encoder for improved cross-lingual generalization.

| MGSM | Avg. | Lrl. | Hrl. |
|---|---|---|---|
| LangBridge | 49.1 | 44.4 | 51.1 |
| LayAlign w/o LLM Input | 51.8 | 45.7 | 54.5 |
| LayAlign | 38.1 | 5.3 | 52.1 |

Table 6: Experiments on MGSM using English-only task data. The complete table of accuracy for each language is in Table 16.

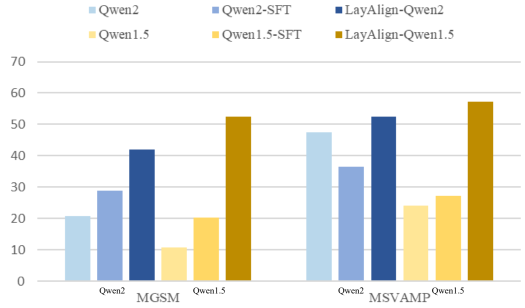


Figure 3: Experimental results for the Swahili language on the MGSM and MSVAMP datasets.

### 5.3 Empowering Multilingual LLM for Low-Resource Languages

In Section 4, we applied LayAlign to both the English-centric LLaMA2 backbone and its multilingual variant, LLaMAX2, which supports all languages evaluated. Here, we further investigate whether LayAlign can empower multilingual LLMs to improve performance in low-resource languages where they underperform. To this end, we utilize the advanced LLMs Qwen1.5-7B-Chat (Bai et al., 2023) and Qwen2-7B-Instruct (Yang et al., 2024), which exhibit strong multilingual capabilities but face challenges in scaling to less-represented languages in their training data. We conduct experiments on the MGSM and MSVAMP benchmarks, focusing on Swahili (Sw), a less-represented language.

Figure 3 presents the results, comparing LayAlign with the vanilla Qwen models and Qwen-SFT fine-tuned on the same multilingual mathematical dataset used in our study. As shown, LayAlign consistently outperforms the baseline methods on both the MGSM and MSVAMP tasks. Comparing vanilla Qwen and Qwen-SFT, we observe that directly fine-tuning these LLMs on multilingual mathematical datasets containing Swahili yields only marginal improvements and, in some cases, even degrades performance. In contrast, LayAlign significantly boosts model performance. On MGSM, LayAlign improves Qwen1.5 and Qwen2 by 41.6 and 21.2 points, respectively, while on MSVAMP, it enhances their performance by 33.2 and 5.1 points, respectively. These results further underscore the potential of our method.

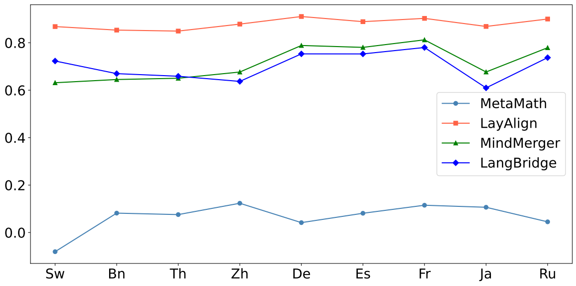


Figure 4: The cosine similarities of the final layer of LLM pooled output representations of English with other languages obtained with the FLORES-101 dataset.

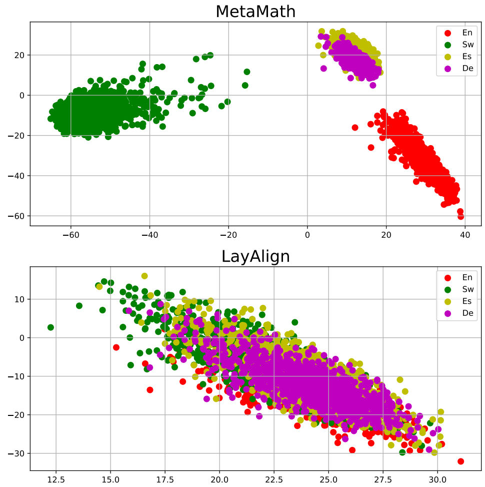


Figure 5: First two principal components of pooled output representations obtained with the FLORES-101.

### 5.4 Analyses of Representation Space

To evaluate whether LayAlign effectively aligns multilingual representations, we compute the cosine similarity between the mean-pooled representations of English and other languages in MGSM, such as Chinese (Zh) and Swahili (Sw), from the final layer of the LLM using the FLORES-101 dataset (Goyal et al., 2022). Figure 4 presents the results for different methods, clearly demonstrating that LayAlign achieves more effective alignment of representations across languages compared to baseline methods. This alignment contributes to the superior performance of LayAlign.

We further illustrate this by visualizing the representations of LayAlign and MetaMath using Principal Component Analysis, as shown in Figure 5. For MetaMath, high-resource languages like Spanish (Es) and German (De) align closely with English (En), while low-resource languages like Swahili

(Sw) are positioned much farther from English. In contrast, LayAlign unifies all languages into a single cluster, indicating more effective alignment of multilingual representations.

## 6 Conclusion

In this paper, we introduce LayAlign, a simple yet effective method designed to leverage multilingual encoders for enhancing the multilingual reasoning capabilities of LLMs. We demonstrate that our approach yields consistent improvements over existing baselines. Notably, LayAlign shows effectiveness in improving cross-lingual reasoning when trained on English-only task data, and LayAlign enables multilingual LLMs to scale to less-represented languages in their training data. Additionally, we provide analyses indicating that LayAlign aligns the representations of various languages with English more effectively. We hope these findings will benefit low-resource language users and inspire further research in this field.

## Limitations

While LayAlign can enhance the performance of English-centric LLMs in low-resource languages through multilingual task training, there remains a performance gap compared to models specifically pretrained and fine-tuned in the target languages.

## Acknowledgements

This project was supported by National Natural Science Foundation of China (No. 62306132, No. 62106138) and Guangdong Basic and Applied Basic Research Foundation. We thank the anonymous reviewers for their insightful feedbacks on this work.

## References

AI@Meta. 2024. Llama 3 model card.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, et al. 2022. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen Marcus McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2024. Llemma: An open language model for mathematics. In *The Twelfth International Conference on Learning Representations*.

Jinze Bai, Shuai Bai, Yunfei Chu, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Jun Chen, Deyao Zhu, Xiaoqian Shen, Xiang Li, Zechun Liu, Pengchuan Zhang, Raghuraman Krishnamoorthi, Vikas Chandra, Yunyang Xiong, and Mohamed Elhoseiny. 2024a. MiniGPT-v2: Large language model as a unified interface for vision-language multi-task learning.

Kaibing Chen, Dong Shen, Hanwen Zhong, Huasong Zhong, Kui Xia, Di Xu, Wei Yuan, Yifei Hu, Bin Wen, Tianke Zhang, Changyi Liu, Dewen Fan, Huihui Xiao, Jiahong Wu, Fan Yang, Size Li, and Di Zhang. 2024b. Evlm: An efficient vision-language model for visual understanding. *Preprint*, arXiv:2407.14177.

Nuo Chen, Zinan Zheng, Ning Wu, Ming Gong, Yangqiu Song, Dongmei Zhang, and Jia Li. 2023a. Breaking language barriers in multilingual mathematical reasoning: Insights and observations. *Preprint*, arXiv:2310.20246.

Nuo Chen, Zinan Zheng, Ning Wu, Ming Gong, Yangqiu Song, Dongmei Zhang, and Jia Li. 2023b. Breaking language barriers in multilingual mathematical reasoning: Insights and observations. *Preprint*, arXiv:2310.20246.

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

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

Tianqing Fang, Zeming Chen, Yangqiu Song, and Antoine Bosselut. 2024. Complex reasoning over logical queries on commonsense knowledge graphs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11365–11384, Bangkok, Thailand. Association for Computational Linguistics.

Sarah E. Finch and Jinho D. Choi. 2024. Convosense: Overcoming monotonous commonsense inferences for conversational ai. *Preprint*, arXiv:2401.15471.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2022. The Flores-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Zixian Huang, Ao Wu, Jiaying Zhou, Yu Gu, Yue Zhao, and Gong Cheng. 2022. Clues before answers: Generation-enhanced multiple-choice QA. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3272–3287, Seattle, United States. Association for Computational Linguistics.

Zixian Huang, Wenhao Zhu, Gong Cheng, Lei Li, and Fei Yuan. 2024. Mindmerger: Efficient boosting llm reasoning in non-english languages. *Preprint*, arXiv:2405.17386.

Bill Yuchen Lin, Seyeon Lee, Xiaoyang Qiao, and Xiang Ren. 2021. Common sense beyond English: Evaluating and improving multilingual language models for commonsense reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1274–1287, Online. Association for Computational Linguistics.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, et al. 2022. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Fenglin Liu, Xuancheng Ren, Guangxiang Zhao, Chenyu You, Xuewei Ma, Xian Wu, and Xu Sun. 2022. Rethinking and improving natural language generation with layer-wise multi-view decoding. *Preprint*, arXiv:2005.08081.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Yinquan Lu, Wenhao Zhu, Lei Li, Yu Qiao, and Fei Yuan. 2024. Llamax: Scaling linguistic horizons of llm by enhancing translation capabilities beyond 100 languages. *arXiv preprint arXiv:2407.05975*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *Preprint*, arXiv:2308.09583.

Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. 2024. Orca-math: Unlocking the potential of slms in grade school math. *Preprint*, arXiv:2402.14830.

Libo Qin, Qiguang Chen, Yuhang Zhou, Zhi Chen, Yinghui Li, Lizi Liao, Min Li, Wanxiang Che, and S Yu Philip. 2025. A survey of multilingual large language models. *Patterns*, 6(1).

Freda Shi, Mirac Suzgun, Markus Freitag, et al. 2023. Language models are multilingual chain-of-thought reasoners. In *The Eleventh International Conference on Learning Representations*.

Oleh Shliazhko, Alena Fenogenova, Maria Tikhonova, Vladislav Mikhailov, Anastasia Kozlova, and Tatiana Shavrina. 2023. mgpt: Few-shot learners go multilingual. *Preprint*, arXiv:2204.07580.

NLLB Team, Marta R. Costa-jussà, James Cross, et al. 2022. No language left behind: Scaling human-centered machine translation. *Preprint*, arXiv:2207.04672.

Qiang Wang, Changliang Li, Yue Zhang, Tong Xiao, and Jingbo Zhu. 2020. Layer-wise multi-view learning for neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4275–4286, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

An Yang, Baosong Yang, Binyuan Hui, et al. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Dongkeun Yoon, Joel Jang, et al. 2024. LangBridge: Multilingual reasoning without multilingual supervision. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7502–7522, Bangkok, Thailand. Association for Computational Linguistics.

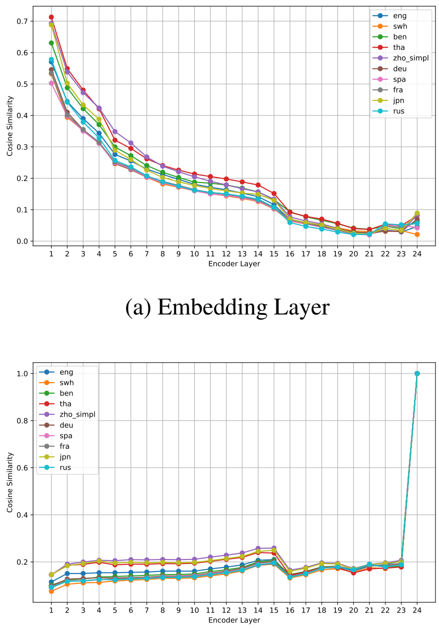
Longhui Yu, Weisen Jiang, Han Shi, et al. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

Fei Yuan, Yinquan Lu, Wenhao Zhu, Lingpeng Kong, Lei Li, Yu Qiao, and Jingjing Xu. 2023. Lego-MT: Learning detachable models for massively multilingual machine translation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11518–11533, Toronto, Canada. Association for Computational Linguistics.

Xiongtao Zhou, Jie He, Yuhua Ke, Guangyao Zhu, Victor Gutierrez Basulto, and Jeff Pan. 2024. An empirical study on parameter-efficient fine-tuning for MultiModal large language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 10057–10084, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

# A Additional Analysis Experiments

## A.1 Analysis of Representation across Layers



(a) Embedding Layer

(b) Last Layer

Figure 6: Cosine similarity. (a) Cosine similarity between the $i$-th encoder layer and the embedding layer in mT5, utilizing the FLORES-101 dataset. (b) Cosine similarity between the $i$-th encoder layer and the last layer in mT5, utilizing the FLORES-101 dataset.

Each encoder layer's representation has different levels of granularity information. As the depth of the encoder layers increases, each layer produces increasingly coarse-grained descriptions of the global context (Liu et al., 2022). As shown in Figure 6, the cosine similarity between the final layer and other encoder layers is markedly different, and the cosine similarity between the $i$-th encoder layer and the embedding layer decreases as $i$ increases. This reflects the shifting granularity of information across different encoder layers. Furthermore, prior studies suggest that intermediate layers can be seen as noisy versions of the final layer's representation, improving model robustness when using layer-wise representations (Wang et al., 2020). Therefore, we use the layer-wise representation of the multilingual encoder to better utilize the language understanding of the encoder to improve the multilingual capabilities of LLM.

To further analyze the representation of different encoder layers, Figure 7 shows the cosine similarity of representations for Chinese and English tokens across the first, twelfth, and final encoder layers. In the first encoder layer, cosine similarity is relatively low, with only token pairs like '五' and 'five,' and '新' and 'new' showing better alignment. By the twenty-fourth layer, many tokens become aligned, yet the similarity between '新' and 'new,' and '地点' and 'site,' is lower than in the twelfth layer. This suggests that the twelfth layer can provide alignment information that supports the final layer. Therefore, utilizing layer-wise representations is crucial for fully leveraging the multilingual capabilities of the encoder.

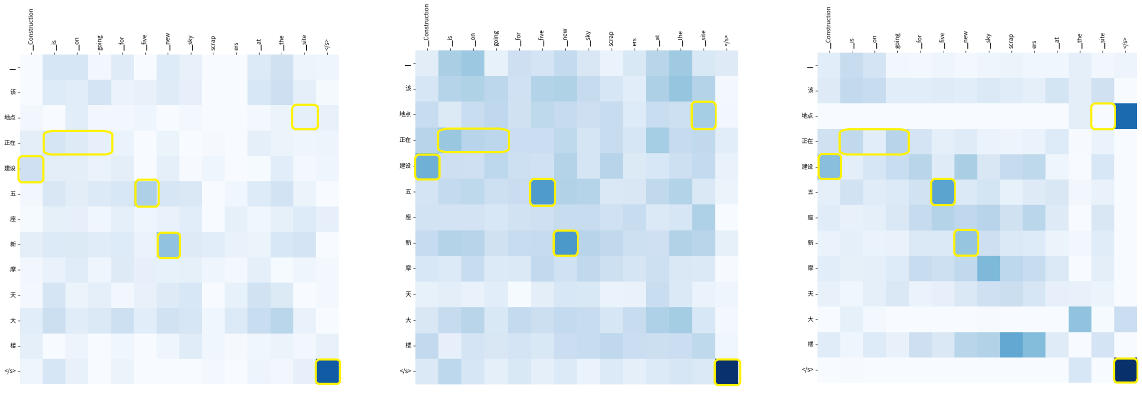
To further assess whether utilizing representations from all layers of the multilingual encoder can enhance the model's multilingual capabilities, we fed the final layer's representation from the multilingual encoder into LayAlign's adaptive fusion-enhanced attention. We conducted experiments on the MGSM dataset, and the results are presented in Table 7. Compared to using only the final layer representation from the multilingual encoder, LayAlign achieved a 1.5% improvement, indicating that representations from other layers of the multilingual encoder also contribute to enhancing the model's multilingual performance.

## A.2 Analysis of Adaptive Fusion-Enhanced Attention

To validate the impact of cross-attention in the adaptive fusion-enhanced attention mechanism, we compute the ratio between the norm of the gate-weighted cross-attention and that of the self-attention across all layers of the LLM, utilizing the FLORES-101 dataset for visualization. As shown in Figure 8, the gate-weighted cross-attention significantly influences the overall attention mechanism, with a more pronounced effect in the deeper layers of the LLM.

We also visualize the layer-wise aligner, as shown in Figure 9. The layer-wise aligner effectively integrates the representations from different layers of the multilingual encoder, providing the LLM with enriched multilingual information by leveraging these fused representations.

LayAlign implements a layer-wise fusion gate that is independent of the current hidden state. To validate this approach, we compare it with a variant called the dynamic gate, where the gate at each layer is determined by the current hidden state. As shown in Table 8, while the dynamic gate yields competitive results, it slightly underperforms compared to the fusion gate in LayAlign. Moreover,



(a) 1th layer of Encoder (b) 12th layer of Encoder (c) 24th layer of Encoder

Figure 7: Visualization of the cosine similarity between representations of Chinese and English sentences in the Encoder.

| MGSM | Avg. | Lrl. | Hrl. | Bn | Th | Sw | Ja | Zh | De | Fr | Ru | Es | En |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LayAlign-LR | 57.5 | 52.9 | 59.5 | 48.8 | 54.0 | 56.0 | 53.2 | 55.6 | 59.6 | 62.0 | 58.8 | 60.4 | 66.8 |
| LayAlign | 59.0 | 56.4 | 60.2 | 51.6 | 59.2 | 58.4 | 52.0 | 56.0 | 62.0 | 61.6 | 61.6 | 61.6 | 66.4 |

Table 7: Experimental results of LayAlign and LayAlign-LR on the MGSM dataset. LayAlign-LR refers to the variant of LayAlign where only the final layer representation from the multilingual encoder is fed into the LLM's adaptive fusion-enhanced attention module.

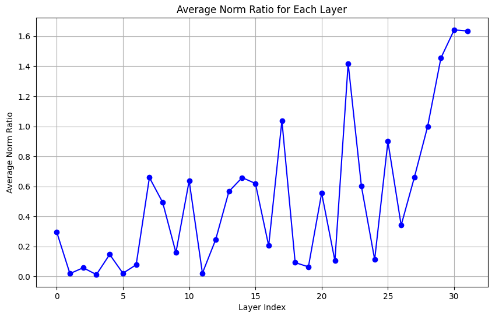


Figure 8: Norm ratio between gate-weighted cross-attention and self-attention across LLM layers using the FLORES-101 dataset. Cross-attention shows a stronger effect in deeper layers.

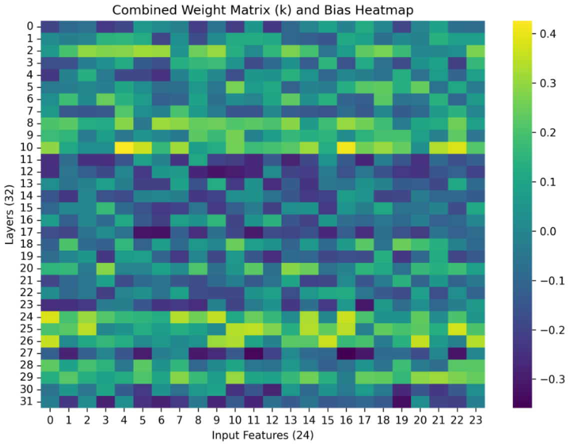


Figure 9: Visualization of the layer-wise aligner

our fusion gate requires fewer gate parameters to be trained, as each layer only requires a single scalar gate parameter.

### A.3 The Analysis of User's Input Text

In our approach, during the translation stage, we use only the output from the adapter as input to the LLM, excluding any user text input. To assess the impact of user text as LLM input during this stage, we conduct experiments on the MGSM dataset. The results are presented in Table 9.

**Stage 1: Translation Stage** The primary objective of Stage 1 is to align the representation space of the multilingual encoder to the LLM through translation-based alignment. As shown in Table 9, it is consistently more effective to omit LLM text input during the translation stage rather than include it.

**Stage 2: Task Stage** In contrast, Stage 2 focuses on leveraging both the multilingual encoder and the LLM for task-specific reasoning. **In multilingual tasks,** including the user's input text in addition to the adapter's output maximizes the LLM's reasoning potential. This configuration is particularly effective for high-resource languages, as the LLM

| MGSM | Avg. | Lrl. | Hrl. | Bn | Th | Sw | Ja | Zh | De | Fr | Ru | Es | En |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LayAlign (Dgate) | 58.8 | 54.5 | 60.6 | 49.6 | 55.6 | 58.4 | 54.4 | 56.4 | 59.2 | 62.0 | 61.2 | 64.4 | 66.8 |
| LayAlign | 59.0 | 56.4 | 60.2 | 51.6 | 59.2 | 58.4 | 52.0 | 56.0 | 62.0 | 61.6 | 61.6 | 61.6 | 66.4 |

Table 8: Experimental results of LayAlign with different gating mechanisms on the MGSM dataset. Dgate denotes dynamic gate.

| Settings | User Input in Stage 1 | User Input in Stage 2 | Avg. | Lrl. | Hrl. |
|---|---|---|---|---|---|
| Multilingual Task Data | | | | | |
| LayAlign + LLM Input at Trans | √ | √ | 57.5 | 50.5 | 60.5 |
| LayAlign - LLM Input at Task | × | × | 56.8 | 55.5 | 57.3 |
| LayAlign | × | √ | **59.0** | **56.4** | **60.2** |
| English Task Data | | | | | |
| LayAlign + LLM Input at Trans | √ | √ | 37.8 | 6.3 | 51.4 |
| LayAlign - LLM Input at Task | × | × | **51.8** | **45.7** | **54.5** |
| LayAlign | × | √ | 38.1 | 5.3 | 52.1 |

Table 9: Experiments of user input at different stages on MGSM. ‘Lrl.’, ‘Hrl.’, and ‘Avg.’ represent the average accuracy across low-resource languages, high-resource languages, and all languages, respectively.

benefits from its existing high-resource knowledge. However, **for English tasks,** including user input in Stage 2 can act as a shortcut, causing the model to rely excessively on the LLM’s inherent English capabilities while neglecting the multilingual encoder. This is reflected in the performance drop observed when user input is included in Stage 2 (e.g., 38.1% compared to 51.8%). The best results are achieved when the LLM is forced to rely on the multilingual encoder rather than directly leveraging its internal English representations.

These results further validate our design choices, demonstrating that LayAlign’s two-stage input strategy effectively balances alignment and reasoning, leading to superior multilingual performance.

## A.4 Analysis of Parameters

The adapter in LayAlign has 25.18M parameters, and the layer-wise aligner contributes 8.39M parameters, resulting in a total of 33.57M trainable parameters. This lightweight design ensures efficiency while maintaining competitive performance. As shown in Figure 10, our experiments demonstrate that lightweight aligners are sufficient for collecting and leveraging information from all encoder layers. Notably, our findings align with the results reported for LangBridge (Yoon et al., 2024), which observed better performance with a simpler Linear adapter compared to a more parameter-intensive MLP design on the XCOPA benchmark (76.6% vs. 72.7%, respectively). This indicates that merely increasing the number of parameters in the aligner does not always yield performance improvements. To evaluate whether larger aligners could enhance performance, we experimented with a modified aligner design, increasing the parameter count from 8.39M to 12.59M by introducing an additional Linear(2048, 2048) layer and a SiLU activation function. However, this modification led to a slight performance drop, with the average accuracy on MGSM decreasing from 59.0 to 58.4. Similarly, increasing the parameters of MindMerger (e.g., from 25.18M to 37.76M) did not result in performance gains, as the accuracy dropped from 57.4 to 57.3. These findings suggest that merely increasing the number of parameters is not a guaranteed path to better performance.

## A.5 Contribution of different Encoder Layers

In this section, we analyze which layers of mT5 contribute most to the performance improvements observed with LayAlign. To investigate this, we conducted experiments with various configurations for the aligner’s input, exploring different ways of extracting representations from the mT5 encoder. Specifically, we tested using (1) the final hidden layer, (2) the mean of all hidden layers, (3) the first 8 layers, (4) the middle 8 layers, and (5) the last 8 layers. The results are presented in Table 11.

Using only the last hidden states of the mT5 encoder led to an average performance drop of 1.5 points compared to LayAlign. This suggests that leveraging hidden states from multiple layers, rather than relying solely on the final layer, enhances the model’s capacity to comprehend multilingual text. The final layer alone appears insufficient for capturing the diverse and hierarchical information encoded across all layers.

Similarly, employing the average hidden states across all layers resulted in a 2.3-point decline in performance compared to LayAlign. This indicates that treating all hidden states as equally important is suboptimal, as it fails to fully exploit the rich linguistic information embedded in the multilingual encoder. In contrast, LayAlign’s adaptive strategy, which dynamically learns individual layer-wise weights, enables the model to prioritize layers based on their relevance to the task. This adaptive weighting mechanism highlights the vary-

| Model | Adapter's Parameters(M) | Layer-Wise Aligner's Parameters(M) | Total Train Parameters (M) | Avg. |
|---|---|---|---|---|
| LayAlign | 25.18 | 8.39 | 33.57 | 59.0 |
| LayAlign + | 25.18 | 12.59 | 37.77 | 58.4 |
| MindMerger | 25.18 | 0 | 25.18 | 57.4 |
| MindMerger+ | 37.76 | 0 | 37.76 | 57.3 |

Table 10: The performance and parameters of models comparison on MGSM. + denotes the experiments with increased parameters.

| MGSM | Avg. | Lrl. | Hrl. | Bn | Th | Sw | Ja | Zh | De | Fr | Ru | Es | En |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Last Hidden States | 57.5 | 52.9 | 59.5 | 48.8 | 54.0 | 56.0 | 53.2 | 55.6 | 59.6 | 62.0 | 58.8 | 60.4 | 66.8 |
| Avgerage Hidden States | 56.7 | 52.5 | 58.5 | 49.6 | 50.4 | 57.6 | 49.2 | 52.4 | 60.4 | 58.0 | 58.4 | 64.4 | 66.8 |
| First 8 Layers | 58.4 | 54.8 | 60 | 51.6 | 54.8 | 58.0 | 51.6 | 55.6 | 60.8 | 59.2 | 60.8 | 63.6 | 68.4 |
| Middle 8 layers | 57.3 | 53.9 | 58.7 | 48.4 | 55.6 | 57.6 | 50.8 | 52.4 | 58.8 | 59.6 | 60.4 | 61.2 | 68.0 |
| Last 8 layers | 57.9 | 54.1 | 59.5 | 52.0 | 52.4 | 58.0 | 50.8 | 54.4 | 63.6 | 58.0 | 56.4 | 65.6 | 68.0 |
| LayAlign | 59.0 | 56.4 | 60.2 | 51.6 | 59.2 | 58.4 | 52.0 | 56.0 | 62.0 | 61.6 | 61.6 | 61.6 | 66.4 |

Table 11: Performance comparison of different mT5 encoder layer selections as inputs to the aligner on MGSM.

ing contributions of different layers in supporting multilingual reasoning.

Moreover, when using hidden states from the first 8 layers, middle 8 layers, and last 8 layers, we observed that the first 8 layers yielded the best performance, while the middle 8 layers performed the worst. This suggests that the middle layers of mT5 contribute relatively less information to the LLM. Since the LLM already integrates the final-layer representation of mT5 through the adapter, incorporating the first 8 layers in the aligner provides additional shallow-layer information, further enriching the LLM's multilingual understanding.

## B Complete Evaluation Results

In this paper, we utilize the following languages, with their respective abbreviations in parentheses. For clarity and ease of reference, these abbreviations are used throughout the text: Bengali (Bn), Thai (Th), Swahili (Sw), Japanese (Ja), Chinese (Zh), German (De), French (Fr), Russian (Ru), Spanish (Es), English (En), Urdu (Ur), Hindi (Hi), Arabic (Ar), Vietnamese (Vi), Polish (Pl), Flemish (Nl), Italian (It), Portuguese (Pt), Turkish (Tr), Greek (El), and Bulgarian (Bg).

Due to space limitations in the main text, the complete results for different languages are provided in this section. Table 12 presents the complete experimental results on the X-CSQA dataset, while Table 13 reports the results on the XNLI dataset. Table 14 illustrates the performance of LayAlign when using different multilingual models as encoders and MetaMath as the LLM on the MGSM dataset. Table 15 provides the ablation study results for LayAlign on the MGSM dataset. Finally, Table 16 shows the experimental results on MGSM using English-only task data.

| X-CQSA | Avg | Ur | Sw | Hi | Ar | Vi | Ja | Pl | Zh | Nl | Ru | It | De | Pt | Fr | Es | En |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LLaMAX2-X-CSQA | 55.0 | 38.9 | 43.1 | 44.3 | 45.5 | 54.1 | 49.4 | 54.6 | 58.1 | 58.5 | 56.9 | 59.1 | 58.9 | 61.1 | 61.4 | 62.7 | 73.9 |
| LLaMAX2-X-CSQA-SFT | 49.4 | 35.4 | 39.2 | 40.0 | 37.8 | 44.0 | 43.9 | 51.8 | 50.5 | 52.9 | 48.7 | 55.8 | 56.1 | 55.1 | 53.4 | 56.6 | 68.6 |
| LangBridge-SFT | 56.7 | 50.6 | 52.5 | 51.6 | 53.6 | 56.4 | 53.1 | 57.6 | 57.8 | 58.2 | 56.0 | 59.6 | 59.2 | 58.8 | 60.4 | 59.4 | 62.4 |
| MindMerger | 61.2 | 50.5 | 51.5 | 51.1 | 54.1 | 60.7 | 55.8 | 64.1 | 64.4 | 64.6 | 61.0 | 64.5 | 64.2 | **65.5** | 64.5 | **67.8** | 75.6 |
| LayAlign | **62.3** | **51.7** | **53.3** | **53.7** | **55.9** | **62.0** | **56.4** | **64.8** | **64.6** | **66.2** | **62.0** | **66.2** | **65.2** | 64.3 | **66.5** | 67.3 | **76.7** |

Table 12: The complete experimental results on X-CSQA datasets. Avg. represents the average accuracy across all languages.

| XNLI | Avg | Sw | Ur | Hi | Th | Ar | Tr | El | Vi | Zh | Ru | Bg | De | Fr | Es | En |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LLaMAX2-XNLI | 76.5 | 66.7 | 65.6 | 70.3 | 66.5 | 73.5 | 71.8 | 76.8 | 77.5 | 78.3 | 80.4 | 81.6 | 82.2 | 83.1 | 84.1 | **89.7** |
| LLaMAX2-XNLI-SFT | 77.4 | 68.3 | 68.3 | 72.1 | 66.7 | 71.7 | 73.2 | 74.3 | 78.5 | 80.3 | 81.9 | 82.7 | 83.7 | **84.7** | **85.1** | 89.3 |
| LangBridge-SFT | 76.0 | 72.2 | **72.2** | 73.4 | **74.3** | 75.0 | 74.5 | 77.2 | 75.4 | 75.9 | 77.1 | 78.2 | 77.4 | 78.0 | 78.5 | 80.8 |
| MindMerger | 79.2 | 72.7 | 71.5 | **74.8** | 73.3 | 77.0 | **76.3** | 78.8 | 80.4 | 80.5 | 80.8 | 82.4 | 83.0 | 84.2 | 84.5 | 88.5 |
| LayAlign | **79.7** | **73.0** | 71.0 | 74.7 | 74.1 | **77.6** | 76.0 | **79.6** | **80.8** | **80.8** | **81.8** | **83.4** | **83.9** | **84.7** | 84.8 | 88.9 |

Table 13: The complete experimental results on XNLI datasets. Avg. represents the average accuracy across all languages.

| MGSM | parm(M) | Avg. | Lrl. | Hrl. | Bn | Th | Sw | Ja | Zh | De | Fr | Ru | Es | En |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| m-GPT | 1418 | 48.5 | 30.8 | 56.1 | 32.0 | 25.6 | 34.8 | 44.8 | 50.0 | 60.0 | 57.2 | 58.0 | 58.0 | 64.4 |
| XGLM | 1733 | 51.1 | 42.4 | 54.8 | 42.4 | 41.2 | 43.6 | 46.0 | 48.0 | 55.6 | 58.0 | 55.2 | 56.8 | 64.0 |
| nllb-3.3B | 1733 | 55.3 | 50.8 | 57.2 | 50.0 | 47.6 | 54.8 | 51.2 | 53.2 | 56.8 | 60.4 | 56.4 | 58.8 | 63.6 |
| mT5-xl | 1670 | 59.0 | 56.4 | 60.2 | 51.6 | 59.2 | 58.4 | 52.0 | 56.0 | 62.0 | 61.6 | 61.6 | 61.6 | 66.4 |

Table 14: LayAlign using different multilingual models as encoder and MetaMath as LLM on the MGSM dataset. Parm(M) represents the number of parameters used in the external model. Lrl., Hrl., and Avg. represent the average accuracy across low-resource languages, high-resource languages, and all languages, respectively.

| MGSM | Avg. | Lrl. | Hrl. | Bn | Th | Sw | Ja | Zh | De | Fr | Ru | Es | En |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| w/o Adapter | 44.1 | 15.9 | 56.2 | 15.2 | 20.8 | 11.6 | 47.2 | 49.2 | 56.4 | 57.6 | 56.8 | 60.4 | 65.6 |
| w/o LLM Input | 56.8 | 55.5 | 57.3 | 50.4 | 59.6 | 56.4 | 49.6 | 54.0 | 56.4 | 60.8 | 60.0 | 58.0 | 62.4 |
| w/o Layer-Wise Aligner | 56.9 | 53.1 | 58.6 | 51.6 | 52.8 | 54.8 | 52.4 | 51.2 | 58.4 | 58.0 | 58.8 | 64.0 | 67.2 |
| w/o Translation Stage | 52.0 | 38.9 | 57.5 | 34.8 | 39.2 | 42.8 | 48.8 | 52.8 | 56.8 | 62.0 | 57.6 | 61.2 | 63.6 |
| w/o Task Stage | 38.8 | 24.7 | 44.9 | 22.8 | 22.8 | 28.4 | 30.8 | 32.0 | 51.6 | 46.4 | 42.8 | 49.6 | 60.8 |
| MetaMath | 37.9 | 5.9 | 51.6 | 6.4 | 6.4 | 4.8 | 34.8 | 39.2 | 56.4 | 55.6 | 51.6 | 55.2 | 68.4 |
| LayAlign | 59.0 | 56.4 | 60.2 | 51.6 | 59.2 | 58.4 | 52.0 | 56.0 | 62.0 | 61.6 | 61.6 | 61.6 | 66.4 |

Table 15: Ablation experiments of LayAlign on the MGSM dataset. Lrl., Hrl., and Avg. represent the average accuracy across low-resource languages, high-resource languages, and all languages, respectively.

| MGSM | Avg. | Lrl. | Hrl. | Bn | Th | Sw | Ja | Zh | De | Fr | Ru | Es | En |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LangBridge | 49.1 | 44.4 | 51.1 | 38.0 | 49.6 | 45.6 | 32.8 | 43.6 | 52.4 | 54.8 | 52.8 | 59.6 | 61.6 |
| LayAlign | 38.1 | 5.3 | 52.1 | 7.2 | 5.2 | 3.6 | 33.2 | 44.8 | 57.2 | 53.2 | 52.4 | 56.8 | 67.2 |
| LayAlign w/o LLM input | 51.8 | 45.7 | 54.5 | 42.0 | 47.2 | 48.0 | 39.6 | 44.4 | 59.2 | 53.2 | 58.8 | 62.4 | 63.6 |

Table 16: Experiments on MGSM using English-only task data. Lrl., Hrl., and Avg. represent the average accuracy across low-resource languages, high-resource languages, and all languages, respectively.