# CM-Align: Consistency-based Multilingual Alignment for Large Language Models

**Xue Zhang**[1,2]*, **Yunlong Liang**[3], **Fandong Meng**[3], **Songming Zhang**[1,2], **Yufeng Chen**[1,2]†, **Jinan Xu**[1,2], **Jie Zhou**[3]

[1]Key Laboratory of Big Data & Artificial Intelligence in Transportation, Beijing Jiaotong University, Ministry of Education

[2]School of Computer Science and Technology, Beijing Jiaotong University, Beijing, China

[3]Pattern Recognition Center, WeChat AI, Tencent Inc, China

{zhang_xue,smzhang22,chenyf,jaxu}@bjtu.edu.cn

## Abstract

Current large language models (LLMs) generally show a significant performance gap in alignment between English and other languages. To bridge this gap, existing research typically leverages the model's responses in English as a reference to select the best/worst responses in other languages, which are then used for Direct Preference Optimization (DPO) training. However, we argue that there are two limitations in the current methods that result in noisy multilingual preference data and further limited alignment performance: 1) Not all English responses are of high quality, and using a response with low quality may mislead the alignment for other languages. 2) Current methods usually use biased or heuristic approaches to construct multilingual preference pairs. To address these limitations, we design a consistency-based data selection method to construct high-quality multilingual preference data for improving multilingual alignment (CM-Align). Specifically, our method includes two parts: consistency-guided English reference selection and cross-lingual consistency-based multilingual preference data construction. Experimental results on three LLMs and three common tasks demonstrate the effectiveness and superiority of our method, which further indicates the necessity of constructing high-quality preference data.

## 1 Introduction

Although existing large language models (LLMs) generally support multiple languages, the performance across different languages is heavily imbalanced (Qin et al., 2024; Xu et al., 2025; Zhang et al., 2025b) and typically shows significantly better general ability in English compared to other languages. The main reason is that English, as the most popular language, benefits from abundant high-quality instruction tuning and preference data (Huang et al., 2025). However, collecting such high-quality multilingual datasets is challenging due to the expensive human annotation and inevitable translation errors (Chen et al., 2024c). To address this problem, existing research (She et al., 2024; Yang et al., 2025c,b) uses the English responses from the model itself as the reference to select the *chosen/rejected* pairs for multilingual data and then utilizes Direct Preference Optimization (Rafailov et al. 2024, DPO) to achieve multilingual alignment.

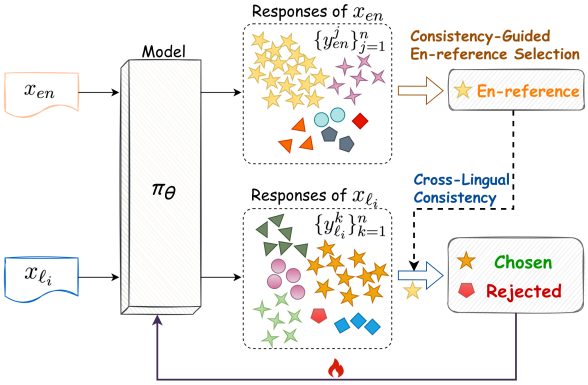


Figure 1: An overview of our method CM-Align. Given a question in different languages, we first request the model to generate multiple responses for each question. Then we select the most consistent one among all English responses as *En-reference*, and calculate the cross-lingual consistency of multiple responses in other languages with *En-reference* to determine the *chosen/rejected* example for DPO optimization.

However, previous work usually treats all English responses as equally valid and utilizes unreliable metrics or heuristic methods to construct multilingual preference pairs, which may introduce non-negligible noise to the preference data and result in limited alignment performance. For example, MAPO (She et al., 2024) uses the translation probability to find the most similar/dissimilar mul-

* This work was done during the internship at Pattern Recognition Center, WeChat AI, Tencent Inc, China.

† Yufeng Chen is the corresponding author.



*Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 25689–25702

tilingual responses with the English responses to construct the *chosen*/*rejected* examples. However, due to the strong semantic constraint, translation probability may not be a suitable metric for quality in scenarios like open-ended generation, math, and code snippets. Additionally, LIDR (Yang et al., 2025c) directly takes the translated answers from English to other languages as the *chosen* examples and the original answers in other languages as the *rejected* examples. This heuristic design may introduce translation errors and translationese into the multilingual preference data. Moreover, CLR (Yang et al., 2025b) applies the self-rewarding method (Chen et al., 2024a) to the multilingual scenario, which requires another aligned LLM to calculate implicit cross-lingual DPO rewards. Although intuitive, English-centric LLMs may not be well-aligned in cross-lingual scenarios and thus offer inaccurate DPO rewards.

To solve these limitations, we design a consistency-based multilingual alignment method (CM-Align), which constructs high-quality multilingual preference data for DPO training (as shown in Figure 1). Our method improves the quality of multilingual preference data from two perspectives: **1)** consistency-guided English reference selection to find the reliable English anchor, and **2)** cross-lingual consistency-based preference data construction with task-specific metrics. Specifically, for each prompt, we first sample multiple responses in English and select the one that is most consistent with others as *En-reference*. Then we choose the responses in other languages that are most consistent with *En-reference* as the *chosen* example and the most inconsistent response as the *rejected* example. Particularly, for different tasks, we design task-specific criteria to better measure the consistency between different responses. Experimental results on Math, Code, and General Instruction Following tasks demonstrate the effectiveness and superiority of our method. Moreover, our method not only shows better alignment performance for in-domain languages but also exhibits excellent generalization for out-of-domain languages, which proves that different languages can learn collaboratively for LLM alignment.

In summary, the major contributions of this paper are as follows[1]:

- We propose a novel multilingual alignment method named CM-Align, which constructs high-quality multilingual preference data based on self-consistency and cross-lingual consistency.
- We design task-specific criteria for evaluating the consistency of any two responses for each prompt, which can be applied in other similar scenarios.
- We evaluate our method on three LLMs and three common tasks, proving the superiority and extensive application of our method.

## 2 Related Work

**Reinforcement Learning from Human Feedback.** RLHF (Christiano et al., 2023; Ziegler et al., 2020; Ouyang et al., 2022) is the critical technique to align LLMs with human preferences and values. Among multiple implementations, Direct Preference Optimization (Rafailov et al. 2024, DPO) streamlines the alignment process by directly optimizing the LLM policy from preference data, which needs fewer computational requirements and shows comparable effectiveness compared to classical RLHF methods. Recently, some works (Azar et al., 2023; Meng et al., 2024; Tang et al., 2024; Hong et al., 2024; Zhang et al., 2025a) modify the training objectives to improve the performance of DPO. Additionally, other works (Deng et al., 2025; Li et al., 2025; Xiao et al., 2025) focus on the data selection method to improve the training performance. However, these methods only concentrate on the alignment for English without considering the challenges of multilingual scenarios.

**Multilingual Alignment.** Currently, many works aim to align the multilingual ability of LLMs to English. Lai et al. (2023) translates English preference data into other languages, which is limited to translation errors. Wu et al. (2024) directly applies the reward model trained in the source language to other target languages. Dang et al. (2024) focuses on exploring data coverage of different languages. Additionally, some works (She et al., 2024; Yang et al., 2025c,b) try to utilize the dominant English to assist the construction of multilingual preference data. MAPO (She et al., 2024) adopts the NLLB model to calculate the translation probability scores of the answers in non-dominant languages and English, and chooses the answer with the highest/lowest score as the *chosen*/*rejected* example. LIDR (Yang et al., 2025c) leverages the

[1]The code is publicly available at https://github.com/XZhang00/CM-Align.

characteristics of translation, *i.e.*, the answers translated with the model itself from English to non-dominant languages as *chosen* examples and the original answers in non-dominant languages as *rejected* examples. CLR (Yang et al., 2025b) adapts the self-rewarding method to the multilingual scenario, which needs another SFT model to calculate implicit cross-lingual rewards. Although these methods have an initial attempt, we argue that the native translation rewards or cross-lingual rewards are unfaithful, further leading to low-quality preference data and underperformed multilingual alignment performance. In this paper, we aim to select high-quality preference data to improve multilingual alignment.

## 3 Methodology

### 3.1 Background

RLHF provides a systematic framework for aligning language models with human preferences through a two-phase pipeline: rewarding modeling and policy optimization.

**Preference-Driven Reward Modeling.** Given a dataset with $N$ samples $\mathcal{D} = \{(x, y_w, y_l)_i\}_N$, where $x$ denotes the input prompt and $y_w/y_l$ represents the human-annotated preferred/dispreferred response, the reward model $r_\phi$ is trained using the Bradley-Terry preference model (Bradley and Terry, 1952). The training objective minimizes the contrastive loss:

$$\mathcal{L}_{\text{RM}}(\phi) = -\mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}}\Big[\log\sigma(r_\phi(x,y_w) - r_\phi(x,y_l))\Big], \quad (1)$$

where $\sigma(\cdot)$ is the sigmoid function that converts reward differences into preference probabilities.

**Policy Optimization.** The learned reward function $r_\phi$ then guides policy optimization through RL algorithms like PPO (Schulman et al., 2017). The objective balances reward maximization against policy divergence:

$$\mathcal{J}_{\text{RLHF}}(\theta) = \max_\theta \mathop{\mathbb{E}}_{\substack{x\sim\mathcal{D}\\ y\sim\pi_\theta(\cdot|x)}}\Big[r_\phi(x,y) - \beta\log\frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}\Big], \quad (2)$$

here $\beta > 0$ controls the Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951) that prevents excessive deviation from the reference policy $\pi_{\text{ref}}$. While effective, this approach requires complex RL implementations and exhibits sensitivity to reward miscalibration.

**Direct Preference Optimization.** As an alternative, DPO (Rafailov et al., 2024) eliminates the separate reward modeling phase through implicit reward parameterization. By establishing a mapping between reward functions and optimal policies, it derives a closed-form solution:

$$r_\theta(x,y) = \beta\log\frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} + \beta\log Z(x), \quad (3)$$

where $Z(x)$ denotes the partition function and independent to $y$ that normalizes the reward distribution. Then the training objective of DPO is derived by substituting Eq. (3) into Eq. (1):

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\Big[\log\sigma\Big(\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta\log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\Big)\Big]. \quad (4)$$

To sum up, DPO streamlines the alignment process by directly optimizing for human preferences, offering superior computational efficiency and stability compared to traditional RLHF approaches, without compromising on alignment quality.

### 3.2 CM-Align

Our approach includes two components: consistency-guided English reference selection and cross-lingual consistency-based construction of multilingual preference data. The key idea is to select a high-quality English response as the reference to guide the choice of the preferred and dispreferred responses for other languages.

Given the model $\pi_\theta$ and a multilingual prompt set $\mathcal{Q} = \{x_{en}, x_{\ell_1}, ..., x_{\ell_k}\}$, where $x_{en}$ denotes an English prompt and $x_{\ell_i}$ represent the parallel prompt in language $\ell_i$, we first request $\pi_\theta$ to generate $n$ responses for each prompt in $\mathcal{Q}$ as the candidate samples.

**Consistency-Guided English Reference Selection.** Given one English prompt $x_{en}$ and the generated $n$ candidate responses $\{y_{en}^j\}_{j=1}^n$ through stochastic sampling from the target model $\pi_\theta$, we select the most consistent response $y_{en}^*$ with each other candidate response as *En-reference*, which is generally a more reliable anchor (Wang et al., 2023).

To judge the consistency of any two responses, we design task-specific criteria for three common types of tasks (Math, Code, and General Instruction Following). Specifically, for responses to Math prompts, we first extract the final numerical answer with regular expressions[2] and conduct majority voting[3] to obtain the *En-reference*. For the responses to Code prompts, we first extract the code snippet and normalize[4] the code snippet, including comment removal, variable anonymization, and code format standardization. Then, given two normalized code snippets, we calculate the weighted average of CodeBLEU (Ren et al., 2020) and CodeBERTScore (Zhou et al., 2023) as the consistency score of any two code responses. Formally, $\text{Cons}_{\text{code}}(c^1_{en}, c^2_{en}) = \alpha \cdot \text{CodeBLEU} + (1-\alpha) \cdot \text{CodeBERTScore}$, where $c^1_{en}$ is the code snippet after the normalization procedures originally extracted from $y^1_{en}$ and $\alpha$ is the hyperparameter to control the weight of CodeBLEU and CodeBERTScore. For the responses to General Instruction Following (GIF) prompts, we first encode each response with the `gte-large-en-v1.5`[5] model to an embedding and then compute the cosine similarity of two embeddings as the consistency score. Formally, $\text{Cons}_{\text{GIF}}(y^1_{en}, y^2_{en}) = \cos(emb(y^1_{en}), emb(y^2_{en}))$, where $emb(y^1_{en})$ is the encoded embedding of $y^1_{en}$.

According to these criteria, we can calculate the consistency score for any two responses, and select the most consistent response $y^*_{en}$ as *En-reference* for each prompt $x_{en}$.

**Cross-Lingual Consistency-based Preference Data Construction.** For each multilingual prompt $x_{\ell_i}$, we first generate $n$ responses $\{y^k_{\ell_i}\}^n_{k=1}$ and then compute the cross-lingual consistency score relative to $y^*_{en}$ for each $y^k_{\ell_i}$, *i.e.*, $\text{CL-Cons}(y^k_{\ell_i}, y^*_{en})$. The cross-lingual consistency score is also task-specific. Specifically, for responses to Math prompts, we also first extract the final numerical answer with regular expressions and judge whether the answer is consistent with the extracted answer of $y^*_{en}$. For responses to Code prompts, we first extract the code snippet from $y^k_{\ell_i}$ and normalize it as $c^k_{\ell_i}$. Then we compute the cross-lingual consistency scores $\text{CL-Cons}_{\text{code}}(y^k_{\ell_i}, y^*_{en}) = \text{Cons}_{\text{code}}(c^k_{\ell_i}, c^*_{en})$, where $c^*_{en}$ is the normalized code extracted from $y^*_{en}$. For the responses to multilingual GIF prompts, we first encode each response with the `gte-multilingual-base`[6] model to an embedding and then compute the cosine similarity between the embedding in other languages and the embedding of $y^*_{en}$ as the cross-lingual consistency score, *i.e.*, $\text{CL-Cons}_{\text{GIF}}(y^k_{\ell_i}, y^*_{en}) = \cos(emb(y^k_{\ell_i}), emb(y^*_{en}))$.

Finally, we select the response with the highest cross-lingual consistency score as the *chosen* example $y^w_{\ell_i}$, and the lowest as the *rejected* example $y^l_{\ell_i}$, *i.e.*, the multilingual DPO training pairs[7] are constructed as:

$$y^w_{\ell_i} = \arg\max_{y^k_{\ell_i}} \text{CL-Cons}(y^k_{\ell_i}, y^*_{en}), \quad (5)$$

$$y^l_{\ell_i} = \arg\min_{y^k_{\ell_i}} \text{CL-Cons}(y^k_{\ell_i}, y^*_{en})). \quad (6)$$

Particularly, we filter out samples that cannot be used to construct *chosen* and *rejected* examples, such as when all the extracted numerical answers or normalized codes are consistent. The filtering strategy guarantees the difference and rationality of DPO training pairs.

**Multilingual Preference Optimization.** Given the multilingual preference dataset $\mathcal{D}_M = \{(x_{\ell_i}, y^w_{\ell_i}, y^l_{\ell_i})\}$, the DPO objective is formulated as:

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(x_{\ell_i}, y^w_{\ell_i}, y^l_{\ell_i}) \sim \mathcal{D}_M} \Big[ \log \sigma \Big( \beta \log \frac{\pi_\theta(y^w_{\ell_i}|x_{\ell_i})}{\pi_{\text{ref}}(y^w_{\ell_i}|x_{\ell_i})} - \beta \log \frac{\pi_\theta(y^l_{\ell_i}|x_{\ell_i})}{\pi_{\text{ref}}(y^l_{\ell_i}|x_{\ell_i})} \Big) \Big]. \quad (7)$$

Additionally, a negative log-likelihood (NLL) loss for the *chosen* examples is incorporated into the vanilla DPO (Rafailov et al., 2024) to improve alignment performance. The NLL objective is as follows:

[2]The regular expressions are listed in Appendix A.

[3]When there are several winners for the math task, we randomly select one as *En-reference*.

[4]The codes for normalization are listed in Appendix B.

[5]The `gte-large-en-v1.5` model is an instruction-tuned multi-lingual embedding model (Li et al., 2023b) released by Alibaba-NLP at https://huggingface.co/Alibaba-NLP/gte-large-en-v1.5.

[6]The `gte-multilingual-base` model achieves state-of-the-art (SOTA) results in multilingual retrieval tasks and multi-task representation model evaluations when compared to models of similar size (Zhang et al., 2024) released by Alibaba-NLP at https://huggingface.co/Alibaba-NLP/gte-multilingual-base.

[7]The constructed data includes the English preference data.

$$\mathcal{L}_{\text{NLL}} = -\mathbb{E}_{(x_{\ell_i}, y^w_{\ell_i})\sim\mathcal{D}_M}\left[\log \frac{\pi_\theta(y^w_{\ell_i}|x_{\ell_i})}{|y^w_{\ell_i}|}\right]. \quad (8)$$

Overall, the training objective $\mathcal{L}$ is: $\mathcal{L} = \mathcal{L}_{\text{DPO}} + \gamma\mathcal{L}_{\text{NLL}}$, where $\gamma$ is a hyper-parameter that controls the weight of $\mathcal{L}_{\text{NLL}}$.

## 4 Experiments

### 4.1 Experimental Setup

In our experiments, we select three multilingual LLMs of different scales: Llama-3.2-3B-Instruct (Grattafiori et al., 2024), Qwen2.5-3B-Instruct (Yang et al., 2025a), and Llama-3-8B-Instruct (AI@Meta, 2024). The three models are all English-centric, *i.e.*, exhibiting suboptimal performance on other languages. Therefore, we aim to utilize the English proficiency to improve the ability of other languages. We conduct main experiments on three common tasks: Math, Code, and General Instruction Following (Zhang et al., 2025c). The benchmark and training data for each task are as follows:

For Math, we use the MGSM (Shi et al., 2022) benchmark to evaluate the mathematical performance of different languages, which involves ten languages: English (*en*), Chinese (*zh*), Spanish (*es*), French (*fr*), Russian (*ru*), Bengali (*bn*), German (*de*), Thai (*th*), Swahili (*sw*), and Japanese (*ja*). Then we randomly select 4.5K English questions (with the ground-truth labels) from MetaMath (Yu et al., 2024) and request `gpt-4o-2024-08-06` to translate[8] English questions to the four languages *zh*, *es*, *fr*, and *ru* as the training data. And the other five languages *bn*, *de*, *th*, *sw*, and *ja* are unseen languages for observing the out-of-domain generalization of each method.

For Code, we use the Multilingual HumanEval (Wang et al., 2024) benchmark to evaluate the code generation ability of different languages, which includes five languages: *en*, *zh*, *es*, *fr*, and *ru*. And we translate[9] the benchmark with `gpt-4o-2024-08-06` to *bn* and *de* as the out-of-domain languages. For the training data, we randomly select 2K examples from the Python subset of Magicoder (Wei et al., 2024) and request `gpt-4o-2024-08-06` to translate English instructions to the four languages *zh*, *es*, *fr*, and *ru*.

For General Instruction Following, we use OMGEval (Liu et al., 2024) as the evaluation benchmark, which is the multilingual version of AlpacaEval (Li et al., 2023a) (including *en*, *zh*, *es*, *fr*, *ru*, *bn*, and *de*) and has undergone the specific localization process and human check. And we also set *bn* and *de* as the out-of-domain languages. We select the `GPT3.5-turbo` model as the baseline comparison model. For the training data, we randomly select 1.5K instructions from Alpagasus (Chen et al., 2024b) and request `gpt-4o-2024-08-06` to translate[10] English instructions to the four languages *zh*, *es*, *fr*, and *ru*.

In all experiments, we generate $n = 30$ responses with temperature 0.5 and top-p 0.9 for each prompt. And the weight of CodeBLEU is set to $\alpha = 0.7$. The other training details are listed in Table 1.

| Settings | Math | | Code | | GIF | |
|---|---|---|---|---|---|---|
| | 3B | 8B | 3B | 8B | 3B | 8B |
| $\beta$ | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $\gamma$ | 1.0 | 1.0 | 0.1 | 0.1 | 0.0 | 0.0 |
| Batch Size | 256 | 256 | 64 | 64 | 16 | 16 |
| Learning Rate | 1e-5 | 1e-6 | 1e-6 | 1e-6 | 1e-6 | 5e-7 |
| Epoch | 3 | 3 | 1 | 1 | 1 | 1 |

Table 1: Detailed training configurations for the three models and three tasks. "3B" denotes Llama-3.2-3B-Instruct and Qwen2.5-3B-Instruct.

### 4.2 Baselines

**SFT-translation.** Only for Math, we request `gpt-4o-2024-08-06` to translate 4.5K English questions and answers to *zh*, *es*, *fr*, and *ru*. And we conduct supervised fine-tuned (SFT) training to explore the alignment performance of translated data. Particularly, the answer to each question in any language is accurate.

**SFT-self-rejection.** Only for Math, we request the model itself to generate 30 responses for each question and conduct rejection sampling for the SFT training, *i.e.*, only select the accurate answer for each question according to the ground-truth label.

**Random-selection.** We randomly select the *chosen*/*rejected* responses for each prompt in any lan-

[8]The translation prompts follow https://github.com/OpenBMB/UltraLink/blob/main/multi-math/math_prompt.yaml.

[9]The translation prompts follow https://github.com/OpenBMB/UltraLink/blob/main/multi-code/code_prompt.yaml.

[10]The translation prompts follow https://github.com/OpenBMB/UltraLink/blob/main/multi-sharegpt/sharegpt_prompt.yaml.

| | In-Domain Languages | | | | | | Out-of-Domain Languages | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Methods** | **en** | **zh** | **es** | **fr** | **ru** | ***ID-avg*** | **bn** | **de** | **th** | **sw** | **ja** | ***OOD-avg*** | ***All-avg*** |
| **Llama-3.2-3B-Instruct** | 71.60 | 59.60 | 67.20 | 60.80 | 62.40 | 64.32 | 44.00 | 63.20 | 58.80 | 51.20 | 49.20 | 53.28 | 58.80 |
| ***with label*** | | | | | | | | | | | | | |
| SFT-translation | 79.20 | 62.80 | 70.40 | 66.80 | 67.60 | 69.36 | 50.40 | 65.20 | 58.00 | 52.80 | 52.80 | 55.84 | 62.60 |
| SFT-self-rejection | 76.00 | 60.80 | 72.00 | 62.80 | 67.60 | 67.84 | 50.00 | 64.40 | **60.80** | 57.20 | 51.20 | 56.72 | 62.28 |
| CM-Align (Ours) | **81.60** | **65.60** | **74.00** | **69.60** | **70.80** | **72.32** | **51.60** | **68.80** | 59.20 | **58.40** | **58.80** | **59.36** | **65.84** |
| ***label-free*** | | | | | | | | | | | | | |
| Random-selection | 72.00 | 60.80 | 68.00 | 60.00 | 60.40 | 64.24 | 48.40 | 61.20 | 54.80 | 52.40 | 48.80 | 53.12 | 58.68 |
| MAPO | 74.40 | 60.40 | 66.80 | 60.00 | 64.80 | 65.28 | 51.20 | 60.40 | 58.00 | 54.40 | 51.60 | 55.12 | 60.20 |
| LIDR | 70.40 | 55.60 | 63.20 | 56.40 | 58.00 | 60.72 | **51.60** | 61.60 | 54.40 | 50.00 | 51.60 | 53.84 | 57.28 |
| CM-Align (Ours) | **78.40** | **63.60** | **72.40** | **68.40** | **68.40** | **70.24** | 50.80 | **67.20** | **60.80** | **61.20** | **60.00** | **60.00** | **65.12** |
| **Qwen2.5-3B-Instruct** | 81.60 | 72.00 | 69.60 | 65.20 | 68.00 | 71.28 | 31.60 | 66.80 | 59.60 | 3.20 | 52.80 | 42.80 | 57.04 |
| ***with label*** | | | | | | | | | | | | | |
| SFT-translation | **82.80** | 72.80 | 66.40 | 65.20 | 63.60 | 70.16 | 28.40 | 65.20 | 56.00 | 5.60 | 51.20 | 41.28 | 55.72 |
| SFT-self-rejection | 78.80 | 70.00 | 68.00 | 64.40 | 74.00 | 71.04 | 35.60 | 66.80 | 58.40 | 5.20 | 56.00 | 44.40 | 57.72 |
| CM-Align (Ours) | **82.80** | **73.60** | **75.20** | **69.60** | **76.00** | **75.44** | **41.20** | **69.60** | **60.00** | **8.00** | **57.60** | **47.28** | **61.36** |
| ***label-free*** | | | | | | | | | | | | | |
| Random-selection | 79.60 | 73.60 | 66.80 | 64.80 | 68.00 | 70.56 | 36.80 | 65.60 | 57.60 | 6.00 | 56.00 | 44.40 | 57.48 |
| MAPO | 80.00 | 73.60 | **72.40** | 65.60 | 72.00 | 72.72 | 37.20 | **68.00** | **64.80** | 6.40 | **60.00** | 47.28 | 60.00 |
| LIDR | 78.80 | 69.20 | 66.40 | 62.40 | 67.60 | 68.88 | 33.20 | 60.80 | 58.80 | 4.00 | 55.20 | 42.40 | 55.64 |
| CM-Align (Ours) | **80.80** | **75.20** | 71.60 | **68.00** | **72.80** | **73.68** | **42.40** | 66.80 | 63.20 | **8.00** | 58.40 | **47.76** | **60.72** |
| **Llama-3-8B-Instruct** | 75.60 | 64.00 | 63.60 | 58.00 | 63.60 | 64.96 | 46.00 | 60.80 | 58.40 | 39.20 | 50.00 | 50.88 | 57.92 |
| ***with label*** | | | | | | | | | | | | | |
| SFT-translation | **80.40** | 59.60 | **73.20** | **66.80** | **68.80** | **69.76** | 40.80 | 65.20 | 60.80 | 39.20 | 51.20 | 51.44 | 60.60 |
| SFT-self-rejection | 75.60 | 62.40 | 66.80 | 58.40 | 65.60 | 65.76 | 46.40 | 62.00 | 57.60 | 34.40 | 50.80 | 50.24 | 58.00 |
| CM-Align (Ours) | 77.20 | **64.80** | 72.00 | 64.00 | 67.60 | 69.12 | **49.20** | **66.80** | **61.20** | **40.80** | **52.80** | **54.16** | **61.64** |
| ***label-free*** | | | | | | | | | | | | | |
| Random-selection | 72.80 | 55.20 | 62.40 | 58.40 | 57.20 | 61.20 | 51.60 | 63.60 | 59.20 | 38.00 | 49.20 | 52.32 | 56.76 |
| MAPO | **76.40** | **66.80** | 65.60 | 58.40 | 59.60 | 65.36 | **55.60** | **65.60** | 56.80 | 42.00 | **54.80** | 54.96 | 60.16 |
| LIDR | 74.40 | 64.00 | 65.60 | 56.00 | 61.60 | 64.32 | 52.00 | 47.60 | 53.20 | 32.00 | 38.80 | 44.72 | 54.52 |
| CM-Align (Ours) | 74.80 | 65.60 | **71.20** | **62.40** | **62.40** | **67.28** | 49.60 | 64.80 | **65.20** | **42.40** | 53.60 | **55.12** | **61.20** |

Table 2: The accuracy (%) results on the MGSM benchmark of the three models. "***with label***" denotes conducting the *En-selection* selection according to the ground-truth label, while "***label-free***" means without access to the ground-truth label. "***ID-avg/OOD-avg***" is the average result of five In-Domain/Out-of-Domain languages and "***All-avg***" is the average result of all ten languages. The result in **bold** means the best result in each setting.

guage.

**MAPO.** MAPO (She et al., 2024) calculates the translation probability between the responses in English and other languages with the `nllb-200-distilled-600M`[11] model, where the responses with the highest/lowest scores are selected as the *chosen*/*rejected* examples.

**LIDR.** LIDR (Yang et al., 2025c) leverages the characteristics of translation to construct preference data, *i.e.*, the answers translated by the model itself from English to non-dominant languages as *chosen* examples and the original answers in non-dominant languages as *rejected* examples. For English, the original generated answers are *chosen* examples, and translated from answers in non-dominant languages to English are *rejected* examples.

[11] https://huggingface.co/facebook/nllb-200-distilled-600M

### 4.3 Main Results

In this section, we present the main results of our method and baselines on MATH, CODE, and General Instruction Following (GIF).

**Results of MATH.** The results on MGSM of Llama-3.2-3B-Instruct, Qwen2.5-3B-Instruct, and Llama-3-8B-Instruct are reported in Table 2, which demonstrate the consistent and significant superiority of our method across multiple languages and models. In both the "*with label*" and "*label-free*" settings, our CM-Align consistently achieves the highest average accuracy (*All-avg*).

In the "*with label*" setting, where *En-reference* is guided by ground-truth labels, our method shows marked improvements over SFT-translation and SFT-self-rejection. SFT-translation only improves performance over the base models slightly, and in some cases, it even leads to a performance degradation (e.g., the *All-avg* of Qwen2.5-3B-Instruct from 57.04% to 55.72%) due to the inevitable translationese. Additionally, SFT-self-rejection also

| | In-Domain Languages | | | | | | Out-of-Domain Languages | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ***label-free* Methods** | **en** | **zh** | **es** | **fr** | **ru** | ***ID-avg*** | **bn** | **de** | ***OOD-avg*** | ***All-avg*** |
| **Llama-3.2-3B-Instruct** | 35.37 | 30.89 | 21.95 | 23.37 | 32.32 | 28.78 | 35.77 | 26.42 | 31.10 | 29.44 |
| Random-selection | 33.54 | 31.91 | 24.39 | 22.15 | 32.93 | 28.98 | 35.77 | 23.17 | 29.47 | 29.12 |
| MAPO | 46.34 | 36.99 | 33.33 | 34.55 | 38.21 | 37.89 | 38.41 | 36.38 | 37.40 | 37.75 |
| LIDR | 22.36 | 22.36 | 5.89 | 13.01 | 22.15 | 17.15 | 17.28 | 9.76 | 13.52 | 16.11 |
| CM-Align (Ours) | **53.66** | **52.03** | **41.67** | **37.20** | **44.92** | **45.89** | **46.14** | **56.10** | **51.12** | **47.39** |
| **Qwen2.5-3B-Instruct** | 65.24 | 52.85 | 34.76 | 27.24 | 42.28 | 44.47 | 45.12 | 58.33 | 51.73 | 46.54 |
| Random-selection | 64.84 | 53.05 | 33.54 | 26.22 | 39.23 | 43.37 | 44.72 | 57.32 | 51.02 | 45.56 |
| MAPO | 69.72 | 55.28 | 36.59 | 30.69 | 42.07 | 46.87 | 43.70 | 58.74 | 51.22 | 48.11 |
| LIDR | 66.26 | 55.89 | **49.80** | 45.53 | 51.83 | 53.86 | 53.46 | 62.20 | 57.83 | 54.99 |
| CM-Align (Ours) | **72.56** | **64.23** | **49.80** | **47.36** | **57.11** | **58.21** | **55.69** | **66.06** | **60.87** | **58.97** |
| **Llama-3-8B-Instruct** | 55.28 | 34.96 | 32.11 | 34.55 | 38.82 | 39.15 | 28.86 | 47.97 | 38.41 | 38.94 |
| Random-selection | 50.00 | 27.64 | 25.20 | 30.69 | 30.08 | 32.72 | 23.98 | 38.41 | 31.20 | 32.29 |
| MAPO | 60.98 | 53.05 | 43.29 | 41.67 | 46.14 | 49.02 | 43.50 | 55.28 | 49.39 | 49.13 |
| LIDR | 61.38 | 42.89 | 35.98 | 37.60 | 34.55 | 42.48 | 38.62 | 45.93 | 42.28 | 42.42 |
| CM-Align (Ours) | **64.43** | **58.54** | **48.17** | **44.72** | **50.61** | **53.29** | **48.58** | **58.13** | **53.35** | **53.31** |

Table 3: The pass@1 (%) results on Multilingual HumanEval of the three models. The result in **bold** means the best result in each language.

brings only a marginal improvement. On the contrary, our method not only achieves the best average performance on in-domain languages but also significantly boosts performance on out-of-domain languages.

The advantages of our CM-Align are even more pronounced in the "*label-free*" setting, which represents a more challenging and realistic application scenario. Without access to any ground-truth labels, our method consistently and substantially outperforms all *label-free* baselines, including Random-selection, MAPO, and LIDR. A key finding is that our method almost reaches the performance of the "*with label*" setting. For example, the *All-avg* 65.12% is close to 65.84% on Llama-3.2-3B-Instruct, and the *All-avg* 61.20% is close to 61.64% on Llama-3-8B-Instruct. As for other baselines, LIDR notably underperforms the original model, particularly in Chinese and French, suggesting this method heavily depends on the translation ability of the model itself, and low-quality preference data leads to performance degradation. Random selection barely improves over the original model, confirming the need for strategic data selection methods. While MAPO demonstrates competitive performance in the *label-free* setting (60.16% on the Llama-3-8B-Instruct model), achieving strong results for English (76.40%) and Bengali (55.60%), it struggles with maintaining consistent improvements across diverse languages and LLMs.

To sum up, our method maintains strong performance for both in-domain and out-of-domain languages, which demonstrates that our method can construct high-quality preference data for better multilingual alignment. Furthermore, the minimal gap between "*with label*" and "*label-free*" settings indicates the practical utility of our approach, enabling better multilingual alignment without requiring expensive labeled data.

**Results of CODE.** Table 3 presents the pass@1 results on Multilingual HumanEval for different alignment methods. The results demonstrate that our method consistently outperforms all other baselines across all languages for all three models. For the Llama-3.2-3B-Instruct model, our CM-Align achieves a remarkable 47.39% average performance, representing a substantial improvement of nearly 18 percentage points over the original model (29.44%). The improvements are particularly pronounced in English (53.66% vs. 35.37%) and Chinese (52.03% vs. 30.89%). Similarly, for Qwen2.5-3B-Instruct and Llama-3-8B-Instruct models, our CM-Align reaches 58.97% and 53.31%, significantly outperforming the original models and other baselines. LIDR exhibits inconsistent performance, struggling significantly with the Llama-3.2-3B-Instruct model (16.11% average) but performing more competitively with Qwen2.5-3B-Instruct (54.99%), which further indicates that the performance of LIDR depends on the translation ability of the original model. Notably, our CM-Align demonstrates strong generalization capabilities, with substantial improvements in out-of-domain languages. These results highlight the quality of our constructed preference data, which can significantly improve code generation capabilities across diverse languages.

| | In-Domain Languages | | | | | | Out-of-Domain Languages | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ***label-free* Methods** | **en** | **zh** | **es** | **fr** | **ru** | ***ID-avg*** | **bn** | **de** | ***OOD-avg*** | ***All-avg*** |
| **Llama-3.2-3B-Instruct** | 64.08 | 14.65 | 27.71 | 24.48 | 14.06 | 29.00 | 47.29 | 43.22 | 45.26 | 33.64 |
| Random-selection | 61.80 | 13.33 | 26.54 | 25.98 | 14.14 | 28.36 | 44.58 | 42.00 | 43.29 | 32.62 |
| MAPO | 65.37 | 24.38 | 33.80 | 29.55 | **17.99** | 34.22 | 52.54 | 52.73 | 52.64 | 39.48 |
| LIDR | **72.90** | 4.01 | 6.98 | 6.09 | 5.46 | 19.09 | 24.43 | 23.49 | 23.96 | 20.48 |
| CM-Align (Ours) | 63.14 | **29.18** | **39.22** | **36.02** | 17.71 | **37.05** | **60.35** | **54.08** | **57.22** | **42.81** |
| **Qwen2.5-3B-Instruct** | 43.23 | 22.20 | 17.84 | 18.45 | 23.49 | 25.04 | 30.47 | 27.67 | 29.07 | 26.19 |
| Random-selection | 42.55 | 30.11 | 18.37 | 19.11 | 27.22 | 27.47 | 29.10 | 31.40 | 30.25 | 28.27 |
| MAPO | 53.34 | 34.47 | 23.80 | 23.92 | 30.54 | 33.21 | 30.36 | 34.09 | 32.23 | 32.93 |
| LIDR | **62.73** | 52.18 | 4.01 | 3.35 | 2.96 | 25.05 | 11.59 | 12.84 | 12.22 | 21.38 |
| CM-Align (Ours) | 46.37 | **55.16** | **30.10** | **30.64** | **38.13** | **40.08** | **35.10** | **47.52** | **41.31** | **40.43** |
| **Llama-3-8B-Instruct** | 63.08 | 22.66 | 36.46 | 38.45 | 36.15 | 39.36 | 59.52 | 52.30 | 55.91 | 44.09 |
| Random-selection | 62.42 | 26.10 | 38.95 | 38.27 | 38.42 | 40.83 | 65.24 | 58.78 | 62.01 | 46.88 |
| MAPO | 65.93 | 33.48 | 46.23 | 40.93 | 38.14 | 44.94 | 71.38 | 61.97 | 66.68 | 51.15 |
| LIDR | **70.80** | 28.84 | 38.12 | 33.97 | 36.38 | 41.62 | 37.15 | 53.29 | 45.22 | 42.65 |
| CM-Align (Ours) | 64.80 | **41.30** | **48.55** | **44.75** | **43.51** | **48.58** | **81.47** | **62.41** | **71.94** | **55.26** |

Table 4: The length-controlled win rate (LCWR, %) results on OMGEval of the three models. The baseline model for comparison is `GPT3.5-turbo`.

| **MATH** | **Llama-3.2-3B** | **Llama-3-8B** | **Qwen2.5-3B** |
|---|---|---|---|
| Ours (*with label*) | 100% | 100% | 100% |
| Random-selection | 13.94% | 12.78% | 9.08% |
| MAPO | 6.69% | 17.19% | 12.90% |
| LIDR | 19.64% | 17.47% | 15.31% |
| Ours (*label-free*) | **91.67%** | **78.85%** | **89.16%** |

Table 5: The average reward accuracy of the 5 in-domain languages.

| **MATH** | **Accuracy** | ***ID-avg*** | ***OOD-avg*** | ***All-avg*** |
|---|---|---|---|---|
| Ground-truth | 100% | 72.32% | 59.36% | 65.84% |
| Ours | **90.84%** | **70.24%** | **60.00%** | **65.12%** |
| Random-En | 80.96% | 67.28% | 56.80% | 62.04% |
| **CODE** | **/** | ***ID-avg*** | ***OOD-avg*** | ***All-avg*** |
| Ours | / | **45.89%** | **51.12%** | **47.39%** |
| Random-En | / | 43.50% | 43.79% | 43.47% |
| **GIF** | **/** | ***ID-avg*** | ***OOD-avg*** | ***All-avg*** |
| Ours | / | **37.05%** | **57.22%** | **42.81%** |
| Random-En | / | 32.06% | 56.55% | 39.05% |

Table 6: The performance degradation of utilizing randomly selected *En-reference* to construct multilingual preference data. "Accuracy" means the accuracy of *En-reference* compared to the ground-truth label (only for MATH), and "**-avg*" represents the average results of Llama-3.2-3B-Instruct.

**Results of GIF.** Table 4 presents the evaluation results on the OMGEval benchmark, comparing the performance against `GPT-3.5-turbo`. We observe several important trends across both model sizes. First, our method CM-Align consistently achieves the best overall performance compared to other baselines. While LIDR excels specifically in English (achieving 72.90%, 62.73%, and 70.80% win rates), it struggles considerably with non-English languages. We guess that the preference data constructed by LIDR is more suitable for improving the English general instruction following capacity, in which scenario, the rewards are more accurate. Additionally, MAPO shows balanced improvements across languages, but doesn't match our method. Our method also demonstrates exceptional generalization to out-of-domain languages, with substantial improvements in Bengali and German. All these results prove the effectiveness of our method again.

## 5 Analysis

### 5.1 Accuracy of Rewards

For the MATH task, we define reward accuracy based on the alignment between selected examples and ground truth. Specifically, an accurate reward occurs when the chosen example's final answer matches the ground truth, while the rejected example's answer differs from it. We show the average reward accuracy of the 5 in-domain languages across different methods and models in Table 5, which demonstrates that our method successfully constructs high-quality preference data with precise reward signals for different models.

### 5.2 Ablation

In this section, we investigate the effectiveness of the consistency-guided selection strategy for *En-reference* and the necessity of designing task-specific consistency metrics for constructing the preference data.

**Random *En-reference*.** We list the performance of using different *En-reference* in Table 6. For

| Criteria | *ID-avg* | *OOD-avg* | *All-avg* |
|---|---|---|---|
| MATH + $\text{Cons}_{\text{math}}$ (Ours) | **70.24%** | **60.00%** | **65.12%** |
| MATH + $\text{Cons}_{\text{GIF}}$ | 62.88% | 52.64% | 57.76% |
| CODE + $\text{Cons}_{\text{code}}$ (Ours) | **45.89%** | **51.12%** | **47.39%** |
| CODE + $\text{Cons}_{\text{GIF}}$ | 35.16% | 34.96% | 35.10% |

Table 7: The results of MATH/CODE with the embedding-based metric ($\text{Cons}_{\text{GIF}}$) for Llama-3.2-3B-Instruct.

MATH, randomly selecting one response as *En-reference* only has 80.96% accuracy, while our consistency-guided selection (voting for Math) strategy can improve the accuracy to 90.84%. The higher accuracy of *En-reference*, the higher accuracy of multilingual preference data. As a result, our method exhibits better performance on the MGSM benchmark in both in-domain and out-of-domain languages compared to the "Random-En" setting and achieves approximate performance with the setting of "Ground-truth" as *En-reference*. For CODE and GIF, our method also outperforms "Random-En". In conclusion, these results demonstrate the effectiveness of our consistency-guided English reference selection strategy for selecting a reliable English anchor to improve multilingual alignment.

**Unify consistency metric for different tasks.** We list the results of MATH/CODE tasks with the embedding-based metric (*i.e.*, $\text{Cons}_{\text{GIF}}$) in Table 7. The results show that even if the *En-reference* is reliable, utilizing an unsuitable metric $\text{Cons}_{\text{GIF}}$ can not select accurate chosen/rejected samples for MATH/CODE to construct high-quality multilingual preference data, resulting in poor alignment performance. Overall, these results prove the necessity and effectiveness of our designed task-specific consistency metrics.

### 5.3 Performance Improvement of Scaling Models

In this section, we explore whether the multilingual alignment performance improves when scaling the model sizes for generating responses. Specifically, we utilize the Qwen2.5-Math-7B/72B-Instruct (Yang et al., 2024) model to generate responses to each question and conduct our method. The results in Figure 2 show that the accuracy of *En-reference* is higher as the model size larger (please refer to the blue line), and the results on MGSM also improve, *e.g.*, "ID-avg" from 70.24 to 73.44/73.84, and "All-avg" from 65.12 to 68.28/67.80. However, the average performance does not improve consistently when the model size scales from 7B to 72B. We suspect that with constrained data volumes (*i.e.*, up to 4.5K samples per language), there exists a performance ceiling that cannot be overcome merely by scaling model sizes. To achieve better multilingual alignment results, the primary focus may be shifted towards expanding the size of the training dataset.

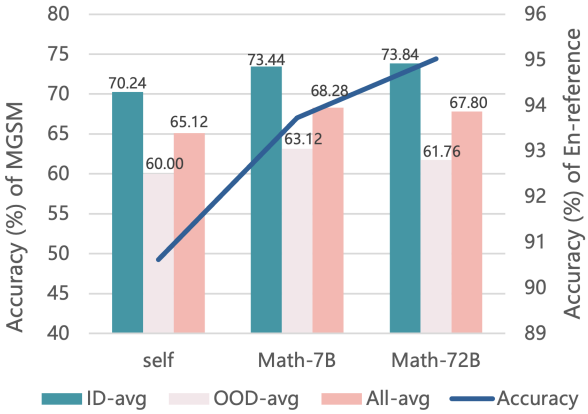


Figure 2: The performance improvement with model size scaling. "Accuracy" (right axis) denotes the accuracy of *En-reference* and "**-avg*" (left axis) represents the average results on MGSM of Llama-3.2-3B-Instruct. "self/Math-7B/72B" means utilizing the Llama-3.2-3B-Instruct/Qwen2.5-Math-7B/72B-Instruct model for generating responses.

## 6 Conclusion

In this paper, we design a consistency-based data selection method to construct high-quality multilingual preference data for improving multilingual alignment. Specifically, our method includes two procedures: consistency-guided English reference selection and cross-lingual consistency-based preference data construction. We conduct extensive experiments on three LLMs with different model sizes and three common tasks (Math, Code, and General Instruction Following). The experimental results demonstrate the effectiveness and superiority of our CM-Align.

## Limitations

To conduct a fast experiment, we do not adopt Iterative DPO, although this training strategy has been proven effective in MAPO (She et al., 2024) and LIDR (Yang et al., 2025c). In the future, we will explore the performance improvement when integrating the Iterative DPO training. Additionally, further experimental investigation is needed to

determine whether expanding the per-task training data volume would lead to enhanced multilingual alignment performance.

## Acknowledgments

The research work described in this paper has been supported by the National Natural Science Foundation of China (No. 62476023, 61976016, 62376019, 61976015), and the authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve this paper.

## References

AI@Meta. 2024. Llama 3 model card.

Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. 2023. A general theoretical paradigm to understand learning from human preferences. *Preprint*, arXiv:2310.12036.

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Changyu Chen, Zichen Liu, Chao Du, Tianyu Pang, Qian Liu, Arunesh Sinha, Pradeep Varakantham, and Min Lin. 2024a. Bootstrapping language models with dpo implicit rewards. *arXiv preprint arXiv:2406.09760*.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2024b. Alpagasus: Training a better alpaca with fewer data. *Preprint*, arXiv:2307.08701.

Nuo Chen, Zinan Zheng, Ning Wu, Ming Gong, Dongmei Zhang, and Jia Li. 2024c. Breaking language barriers in multilingual mathematical reasoning: Insights and observations. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7001–7016, Miami, Florida, USA. Association for Computational Linguistics.

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2023. Deep reinforcement learning from human preferences. *Preprint*, arXiv:1706.03741.

John Dang, Arash Ahmadian, Kelly Marchisio, Julia Kreutzer, Ahmet Üstün, and Sara Hooker. 2024. RLHF can speak many languages: Unlocking multilingual preference optimization for LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13134–13156, Miami, Florida, USA. Association for Computational Linguistics.

Xun Deng, Han Zhong, Rui Ai, Fuli Feng, Zheng Wang, and Xiangnan He. 2025. Less is more: Improving llm alignment via preference data selection. *Preprint*, arXiv:2502.14560.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, and et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. *Preprint*, arXiv:2403.07691.

Kaiyu Huang, Fengran Mo, Xinyu Zhang, Hongliang Li, You Li, Yuanchi Zhang, Weijian Yi, Yulong Mao, Jinchen Liu, Yuzhuang Xu, Jinan Xu, Jian-Yun Nie, and Yang Liu. 2025. A survey on large language models with multilingualism: Recent advances and new frontiers. *Preprint*, arXiv:2405.10936.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Viet Dac Lai, Chien Van Nguyen, Nghia Trung Ngo, Thuat Nguyen, Franck Dernoncourt, Ryan A. Rossi, and Thien Huu Nguyen. 2023. Okapi: Instruction-tuned large language models in multiple languages with reinforcement learning from human feedback. *Preprint*, arXiv:2307.16039.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023a. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`.

Yisen Li, Lingfeng Yang, Wenxuan Shen, Pan Zhou, Yao Wan, Weiwei Lin, and Dongping Chen. 2025. Crowdselect: Synthetic instruction data selection with multi-llm wisdom. *Preprint*, arXiv:2503.01836.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023b. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Yang Liu, Meng Xu, Shuo Wang, Liner Yang, Haoyu Wang, Zhenghao Liu, Cunliang Kong, Yun Chen, Yang Liu, Maosong Sun, and Erhong Yang. 2024. Omgeval: An open multilingual generative evaluation benchmark for large language models. *Preprint*, arXiv:2402.13524.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. Simpo: Simple preference optimization with a reference-free reward. *Preprint*, arXiv:2405.14734.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Libo Qin, Qiguang Chen, Yuhang Zhou, Zhi Chen, Yinghui Li, Lizi Liao, Min Li, Wanxiang Che, and Philip S. Yu. 2024. Multilingual large language model: A survey of resources, taxonomy and frontiers. *Preprint*, arXiv:2404.04925.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

Shuo Ren, Daya Guo, Shuai Lu, Long Zhou, Shujie Liu, Duyu Tang, Neel Sundaresan, Ming Zhou, Ambrosio Blanco, and Shuai Ma. 2020. Codebleu: a method for automatic evaluation of code synthesis. *Preprint*, arXiv:2009.10297.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shuaijie She, Wei Zou, Shujian Huang, Wenhao Zhu, Xiang Liu, Xiang Geng, and Jiajun Chen. 2024. MAPO: Advancing multilingual reasoning through multilingual-alignment-as-preference optimization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10015–10027, Bangkok, Thailand. Association for Computational Linguistics.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. 2022. Language models are multilingual chain-of-thought reasoners. *Preprint*, arXiv:2210.03057.

Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Rémi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Ávila Pires, and Bilal Piot. 2024. Generalized preference optimization: A unified approach to offline alignment. *Preprint*, arXiv:2402.05749.

Haoyu Wang, Shuo Wang, Yukun Yan, Xujia Wang, Zhiyu Yang, Yuzhuang Xu, Zhenghao Liu, Liner Yang, Ning Ding, Xu Han, Zhiyuan Liu, and Maosong Sun. 2024. UltraLink: An open-source knowledge-enhanced multilingual supervised fine-tuning dataset. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11929–11942, Bangkok, Thailand. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. 2024. Magicoder: Empowering code generation with oss-instruct. *Preprint*, arXiv:2312.02120.

Zhaofeng Wu, Ananth Balashankar, Yoon Kim, Jacob Eisenstein, and Ahmad Beirami. 2024. Reuse your rewards: Reward model transfer for zero-shot cross-lingual alignment. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1332–1353, Miami, Florida, USA. Association for Computational Linguistics.

Yao Xiao, Hai Ye, Linyao Chen, Hwee Tou Ng, Lidong Bing, Xiaoli Li, and Roy Ka wei Lee. 2025. Finding the sweet spot: Preference data construction for scaling preference optimization. *Preprint*, arXiv:2502.16825.

Yuemei Xu, Ling Hu, Jiayi Zhao, Zihan Qiu, Kexin Xu, Yuqi Ye, and Hanwen Gu. 2025. A survey on multilingual large language models: Corpora, alignment, and bias. *Frontiers of Computer Science*, 19(11):1911362.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Qwen: An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2025a. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Wen Yang, Junhong Wu, Chen Wang, Chengqing Zong, and Jiajun Zhang. 2025b. Implicit cross-lingual rewarding for efficient multilingual preference alignment. *Preprint*, arXiv:2503.04647.

Wen Yang, Junhong Wu, Chen Wang, Chengqing Zong, and Jiajun Zhang. 2025c. Language imbalance driven rewarding for multilingual self-improving. *Preprint*, arXiv:2410.08964.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. *Preprint*, arXiv:2309.12284.

Songming Zhang, Xue Zhang, Tong Zhang, Bojie Hu, Yufeng Chen, and Jinan Xu. 2025a. Aligndistil: Token-level language model alignment as adaptive policy distillation. *Preprint*, arXiv:2503.02832.

Xin Zhang, Yanzhao Zhang, Dingkun Long, Wen Xie, Ziqi Dai, Jialong Tang, Huan Lin, Baosong Yang, Pengjun Xie, Fei Huang, and 1 others. 2024. mgte: Generalized long-context text representation and reranking models for multilingual text retrieval. *arXiv preprint arXiv:2407.19669*.

Xue Zhang, Yunlong Liang, Fandong Meng, Songming Zhang, Yufeng Chen, Jinan Xu, and Jie Zhou. 2025b. Less, but better: Efficient multilingual expansion for LLMs via layer-wise mixture-of-experts. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 17948–17963, Vienna, Austria. Association for Computational Linguistics.

Xue Zhang, Songming Zhang, Yunlong Liang, Fandong Meng, Yufeng Chen, Jinan Xu, and Jie Zhou. 2025c. A dual-space framework for general knowledge distillation of large language models. *Preprint*, arXiv:2504.11426.

Shuyan Zhou, Uri Alon, Sumit Agarwal, and Graham Neubig. 2023. Codebertscore: Evaluating code generation with pretrained models of code. *Preprint*, arXiv:2302.05527.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences. *Preprint*, arXiv:1909.08593.

## A Extraction Regular Expressions for Math responses

For the responses to Math prompts, we first utilize the following code to extract the final numerical value as the answer for judging the consistency of any two responses.

```
def extract_last_num(text: str) -> float:
    text = re.sub(r"(\d),(\d)", "\g<1>\g<2>", text) # processing for 123,456
    res = re.findall(r"(\d+(\.\d+)?)", text) # matching for 123456.789
    if len(res) > 0:
        num_str = res[-1][0]
        return float(num_str)
    else:
        return 0.0
```

## B Codes for Normalizing Code Snippets

For the responses to Code prompts, we first extract the code snippet and utilize the following code to normalize the code snippet.

```
class CodeNormalizer:
    def __init__(self,
                 remove_comments=True,
                 anonymize_variables=True,
                 standardize_format=True):
        self.remove_comments = remove_comments
        self.anonymize_variables = anonymize_variables
        self.standardize_format = standardize_format

    def process(self, code):
        """Carry out the complete code normalization process."""
        if self.remove_comments:
            code = self._remove_comments(code)
        if self.anonymize_variables:
            code = self._anonymize_variables(code)
        if self.standardize_format:
            code = self._standardize_format(code)
        return code

    def _remove_comments(self, code):
        """Remove all comments (including inline comments) and retain the code structure."""
        try:
            io_obj = StringIO(code)
            tokens = list(tokenize.generate_tokens(io_obj.readline))
            filtered_tokens = [t for t in tokens if t.type != tokenize.COMMENT]
            new_code = tokenize.untokenize(filtered_tokens)
            return new_code.decode('utf-8').replace('\r\n', '\n').replace('\r', '\n')
        except Exception as e:
            print(f"_remove_comments error: {e}")
            return code

    def _anonymize_variables(self, code):
        """Variable anonymization for maintaining scope consistency."""
        class Renamer(ast.NodeTransformer):
            def __init__(self):
                self.var_map = {}
                self.counter = 0

            def visit_Name(self, node):
                if isinstance(node.ctx, ast.Store):
                    if node.id not in self.var_map:
                        self.var_map[node.id] = f"var{self.counter}"
                        self.counter += 1
                if node.id in self.var_map:
                    return ast.Name(id=self.var_map[node.id], ctx=node.ctx)
                return node
        try:
            tree = ast.parse(code)
            tree = Renamer().visit(tree)
```

```
            return astor.to_source(tree)
        except Exception as e:
            print(f"_anonymize_variables error: {e}")
            return code

    def _standardize_format(self, code):
        """Code format standardization."""
        try:
            import black
            return black.format_str(code, mode=black.FileMode())
        except ImportError:
            try:
                tree = ast.parse(code)
                return astor.to_source(tree)
            except:
                return code
        except Exception as e:
            print(f"_standardize_format error: {e}")
            return code
```

---