Synthetic Data for Reinforcement Learning from Human Feedback

A Comparative Analysis of Synthetic Data Architectures for Multilingual Alignment

Pascal Mathas

Bachelor thesis
Credits: 18 EC

Bachelor *Kunstmatige Intelligentie*

University of Amsterdam
Faculty of Science
Science Park 900
1098 XH Amsterdam

Supervisor
Redacted & Redacted

Informatics Institute
Faculty of Science
University of Amsterdam
Science Park 900
1098 XH Amsterdam

Semester 2, 2024-2025

Abstract

In this paper, we compare three architectures for creating a synthetic multilingual preference dataset. All architectures are based on a custom-created English DPO dataset. In Architecture A, we translate everything (prompt, chosen, and rejected responses) into the target language using the same model. In Architecture B, we translate only the prompt and generate the chosen and rejected responses using models of differing quality (high-quality for chosen, low-quality for rejected). In Architecture C, we translate only the prompt and generate only the chosen responses, while the rejected responses are translated from English using a lower-quality model to intentionally introduce translation artifacts. We find that Architectures B and C, even with very little data, improve the multilingual performance of the Aya-23-8B model, whereas Architecture A does not. The primary reason is that B and C provide the model with more natural, in-target-language-generated chosen responses, thereby avoiding English biases.

Table of Contents

1 Introduction **4**

2 Related works **5**

3 Method **5**
3.1 Datasets . . . 5
3.2 Target Languages . . . 6
3.3 Assessing Translation Quality . . . 7
3.4 Architecture A . . . 7
3.5 Architecture B . . . 7
3.6 Architecture C . . . 7
3.7 Direct Preference Optimization . . . 7
3.8 Evaluation . . . 8

4 Experiments **8**
4.1 Assessing Translation Quality . . . 8
4.1.1 WMT24++ . . . 8
4.1.2 Models . . . 8
4.1.3 Metrics . . . 9
4.2 Architecture A . . . 9
4.3 Architecture B . . . 9
4.4 Architecture C . . . 10
4.5 Direct Preference Optimization . . . 10
4.6 Evaluation . . . 10
4.7 Ablations . . . 10
4.8 Computational Requirements . . . 11

5 Results **11**
5.1 Assessing Translation Quality . . . 11
5.2 Direct Preference Optimization . . . 13
5.3 Ablations . . . 14
5.3.1 Less Tasks . . . 14
5.3.2 Less Languages . . . 15

6 Discussion **16**
6.1 DPO Averages . . . 16
6.2 DPO per Language . . . 16
6.3 Ablations . . . 16
6.4 Architecture . . . 17
6.5 Limitations . . . 17

7 Conclusion **17**

A WMT24++ **20**
A.1 Downsampling . . . 20
A.2 Model Summaries . . . 21

B Evaluation **21**
B.1 Judge Prompt . . . 21

C Direct Preference Optimization **22**
C.1 Results with Standard Deviation . . . 22
C.2 Per-Language Win-Loss Rates for Aya-A . . . 22
C.3 Per-Language Win-Loss Rates for Aya-B . . . 22
C.4 Per-Language Win-Loss Rates for Aya-C . . . 23

1 Introduction

Preference optimization through Reinforcement Learning from Human Feedback (RLHF) has proven beneficial for training large language models (LLMs) (Bai, Jones, et al. 2022; Bai, Kadavath, et al. 2022; Christiano et al. 2017; Ouyang et al. 2022). However, this has primarily been applied to common languages such as English and Chinese, since most preference optimization datasets are available only in these languages (Dang et al. 2024). To improve the alignment of LLMs with humans in more diverse and less common languages, these models should be fine-tuned through RLHF using multilingual data, thereby making them more effective in varied linguistic contexts.

Preference optimization through RLHF aims to align LLMs with human goals. This is achieved by fine-tuning LLMs using human-annotated preference data, which consists of a prompt and multiple responses. One of the responses is preferred for being more aligned with human values, morals, and goals. This response receives a higher reward signal from the reward model, steering the LLM toward more human-like behavior. This algorithm is known as Proximal Policy Optimization (PPO) (Schulman et al. 2017). A newer algorithm that seeks to eliminate the unstable reward model in PPO is Direct Preference Optimization (DPO). DPO bypasses the reward model by incorporating the preference loss directly into the policy function, and it has proven to be just as effective in aligning LLMs with humans (Rafailov et al. 2023).

A key issue with both methods is the high computational cost of obtaining human-annotated preference data, particularly in multiple languages, where such data is sparse and often limited to English or Chinese. One solution is to generate synthetic preference data, thereby reducing the need for manual annotation. However, creating a multilingual synthetic dataset presents several challenges, including selecting which languages and tasks to include, ensuring task diversity, and choosing an effective generation strategy, such as translating an existing English dataset into multiple languages or translating prompts and generating responses directly in each target language.

As many of these challenges remain unexplored, this represents a highly relevant and important area of research. In this paper, we therefore explore one of these challenges, namely, which architecture is best to use. We examine three architectures, all of which can be used to create a preference dataset. Each architecture utilizes our custom-created English preference dataset.

- **(A) P-trans and R-trans.** In this architecture, we translate our English preference optimization dataset into x target languages. This means using the same translation model to translate the prompt, the chosen response, and the rejected response.
- **(B) P-trans and R-gen.** In this architecture, we translate only the prompts from our English preference optimization dataset into x target languages. We then generate the chosen and rejected responses in the target languages. To better differentiate the two responses, we use a higher-quality model for the chosen responses and a lower-quality model for the rejected responses.
- **(C) R-gen and R-trans.** In this architecture, we again translate only the prompts from our English preference optimization dataset into x target languages. We then generate only the chosen responses in the target languages. The English rejected responses from the original dataset are translated into the target languages. This architecture differentiates the chosen and rejected responses by introducing translation artifacts into the rejected responses. It is adapted from Dang et al. (2024).

Each of the architectures will be evaluated by first fine-tuning the multilingual, open-source Aya-23-8B model (Aryabumi et al. 2024), and then comparing its performance against three baselines. The main research question of this paper is therefore as follows: "How do the proposed architectures (P-trans/R-trans, P-trans/R-gen, and R-gen/R-trans) compare in their effectiveness for creating multilingual synthetic preference datasets when used to fine-tune an LLM via Direct Preference Optimization?"

We find that architectures B and C, even with very little data, manage to improve the multilingual performance of the Aya-23-8B model, whereas architecture A does not. The primary reason for this is that B and C provide the model with clearly distinguishable, lower-quality negative examples, unlike architecture A.

2 Related works

Direct Preference Optimization. RLHF is used to optimize LLMs toward more human-like responses. PPO attempts to align the LLM by using a reward model to assign scores to the responses generated by the model (Schulman et al. 2017). The reward model is therefore an essential component of this method, but it must be trained and is often unstable in its predictions. Rafailov et al. (2023) introduce DPO, which aims to improve the alignment process by avoiding reinforcement learning altogether. Instead, it directly optimizes the LLM using a classification loss during training. Although this method does not rely on a reward model, it still achieves competitive performance. Later variations, such as Iterative DPO (Yuan et al. 2025), further eliminate the need for a preference dataset, as the LLM generates and rewards its own responses through iterative self-improvement.

Multilingual Optimization. Although we focus in this paper on optimizing LLMs in a multilingual context via a synthetic preference dataset, alternative approaches have also been explored to bypass the creation of such datasets altogether. For instance, She et al. (2024) propose a framework called Multilingual-Alignment-as-Preference Optimization (MAPO), where they align non-English responses to the often superior English responses produced by the model. Another framework, West-of-N, introduced by Pace et al. (2024), generates synthetic preference data by leveraging the outputs from Best-of-N sampling. It constructs preference pairs by selecting the best and worst responses from the generated samples. Both methods eliminate the explicit need to create a multilingual preference dataset, but are therefore more complex and harder to implement.

Multilingual Preference Data. Very little has been written about the creation of multilingual synthetic datasets, which aim to improve the multilingual performance of LLMs. One early work by Lai et al. (2023) constructs such a dataset by using ChatGPT to translate an English dataset into various target languages. A major limitation of this approach is its heavy reliance on translations, which results in poor data quality (Dang et al. 2024). A more recent study by Dang et al. (2024) seeks to improve the quality of multilingual synthetic datasets by generating multiple responses directly in the target languages, thereby avoiding translation artifacts. We also adapted this pipeline as Architecture C in this paper. Although Dang et al. (2024) demonstrate that their architecture improves the multilingual performance of the model, the study is not without its limitations. The most notable limitation is the lack of transparency, as many exact details of the pipeline remain unclear. Additionally, no code has been made available, and the dataset used is no longer publicly accessible.

3 Method

In this section, we explain the basic methodology for creating the synthetic dataset, assessing the translation quality of different translation models, and building and evaluating the three architectures proposed in this paper.

3.1 Datasets

We construct a base English preference dataset that serves as the foundation for the three architectures. To ensure that the dataset includes a variety of tasks, we combine five existing preference datasets, each of which focuses on a different task (Table 1). For all datasets, we follow the same pre-processing procedure. First, we filter out any multi-turn, conversation-style entries, so that the datasets consist only of single-turn, question-answering-style prompts and responses. We then filter the datasets to a maximum length of 2,048 tokens due to computational limitations for translation and generation. Once again, due to computational limitations, we only select 250 random entries from each dataset. However, as Zhou et al. (2023) indicated, high-quality data is more important than quantity. The final English preference dataset therefore contains 1,250 entries.

Dataset Name	Task Description	Dataset Size	Used Size
	Human Labeled		
Chatbot Arena	LLM Evaluation	33,000	250
Nvidia/HelpSteer2	Helpfulness Preference	21,362	250
Tasksource/Dpo-pairs	Discriminative Tasks	5,128,939	250
	AI Labeled		
Gutenberg-DPO	Novel Writing	918	250
Math-Step-DPO	Mathematical Reasoning	10,795	250

Table 1: Overview of selected DPO datasets, categorized by labeling source. Table based on Xiao et al. 2024.

Chatbot Arena. The Chatbot Arena dataset (Zheng et al. 2023) contains question-and-answer-style prompts and responses spanning a variety of tasks. Responses are generated by different models, and the preferred response is selected by humans.

HelpSteer2. The HelpSteer2 dataset (Wang, Bukharin, et al. 2024; Wang, Dong, et al. 2024) similarly consists of a variety of prompts spanning multiple tasks. Compared to the Chatbot Arena dataset, the prompts and responses are not question-answering style, but can be single words or phrases. Responses are assessed and ranked on multiple criteria by humans. For the chosen response, we take the one with the highest average score across the five helpfulness criteria.

Tasksource The Tasksource dataset (Sileo 2024) contains discriminative tasks, where the prompt consists of a two-part statement, and the responses indicate whether the statement is, for instance, true, false, or contradictory. All entries are human-annotated.

Gutenberg. The Gutenberg dataset (Jondurbin 2024) is the first of the two AI-labeled datasets. It contains either novel-writing or summarization tasks, using famous books, such as Huckleberry Finn as its source. The rejected response is the text generated by the LLM, while the chosen response is the actual chapter from the book.

Math-Step. The Math-Step dataset (Xinlai 2024) is the second AI-labeled dataset and consists of mathematical reasoning questions. As the dataset was originally created to boost the mathematical reasoning capabilities of LLMs with Step-DPO, each answer includes initial reasoning steps. To better differentiate the chosen and rejected responses, we use only the final LLM-generated step for both.

3.2 Target Languages

Code	Lang2LP	Language	Script	Family	Subgrouping
ar	en-ar_EG	Arabic	Arabic	Afro-Asiatic	Semitic
zh	en-zh_CN	Chinese	Han & Hant	Sino-Tibetan	Sinitic
de	en-de_DE	German	Latin	Indo-European	Germanic
is	en-is_IS	Icelandic	Latin	Indo-European	Germanic
ja	en-ja_JP	Japanese	Japanese	Japonic	Japanesic
nl	en-nl_NL	Dutch	Latin	Indo-European	Germanic
ru	en-ru_RU	Russian	Cyrillic	Indo-European	Balto-Slavic
es	en-es_MX	Spanish	Latin	Indo-European	Italic

Table 2: Selected target languages for the experiments of this paper. **code**: language identifying code. **lang2lp**: English to target language translation code. Table based on Dang et al. (2024).

We selected eight target languages for both the assessment of translation quality and the creation of the synthetic multilingual preference dataset. These languages are German, Dutch, Ice-

landic, Spanish, Arabic, Chinese, Japanese, and Russian. We chose these languages because they span a wide range of language families, scripts, and sub-groupings, as shown in Table 2. After executing each architecture, the preference optimization dataset is expanded to nine languages (including English) and has a final size of 11,250 entries.

3.3 Assessing Translation Quality

Besides the dataset, another main component of each architecture is the translation model. Although architectures B and C also include a generative component, they still rely on translated prompts from the English preference optimization dataset. We therefore assess a variety of models based on their translation quality. Based on their performance, we select the best-performing, computationally viable model for the base translations.

3.4 Architecture A

For this architecture, we translate the English preference dataset into the eight target languages. This means that the prompt, chosen response, and rejected response are all translated by the same model. For the translation model, we select the best-performing model from the translation benchmark. The theoretical motivation behind this architecture is to preserve the original high-quality English preference data by using a high-quality translation model.

3.5 Architecture B

In this architecture, we translate only the prompts from the English preference dataset into the eight target languages. We then generate both the chosen and rejected responses in the target languages. To differentiate the response pair, we use a larger, high-quality model for the chosen responses and a smaller, lower-quality model for the rejected responses. The theoretical motivation behind this architecture is to avoid translation artifacts, thereby steering the model toward more natural, human-like responses.

3.6 Architecture C

For architecture C, we again translate only the prompts from the English preference dataset into the eight target languages using the best-performing translation model. We then use the same high-quality generated chosen responses as in architecture B. For the rejected responses, we translate the English rejected responses into the eight target languages using the worst-performing translation model. The theoretical motivation behind this approach is to explicitly introduce translation artifacts into the rejected responses, thereby aligning the model with more natural, human-like responses.

3.7 Direct Preference Optimization

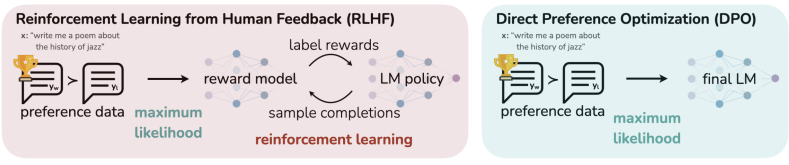

Figure 1: Differences in the pipeline between RLHF and DPO. DPO omits the need to train a reward model by directly optimizing the LLM with a classification objective. This, in turn, removes any unstable reward function or reinforcement learning component from the pipeline. This figure is adapted from Rafailov et al. (2023).

We use our three synthetic multilingual preference datasets to fine-tune a multilingual LLM. We choose to fine-tune the LLM using Direct Preference Optimization (DPO) instead of RLHF, as it offers several advantages. The primary benefit is that it removes the unstable reinforcement

learning component associated with the reward model (Rafailov et al. 2023). Figure 1 illustrates the differences between RLHF and DPO. By removing the reward model and directly optimizing the LLM with a classification objective, the unstable elements of RLHF are eliminated from the pipeline. This results in more stable and predictable learning (Rafailov et al. 2023).

3.8 Evaluation

To evaluate each of the architectures, we fine-tune a multilingual-optimized model using the preference datasets created by each architecture. After fine-tuning, we generate responses for a multilingual evaluation dataset and calculate the win rates of the fine-tuned model against three state-of-the-art baseline models.

4 Experiments

In this section, we outline our experimental setup in detail, including the specific models, metrics, libraries, and parameters used. Figure 2 gives an overview of each of the architectures and the complete experimental setup.

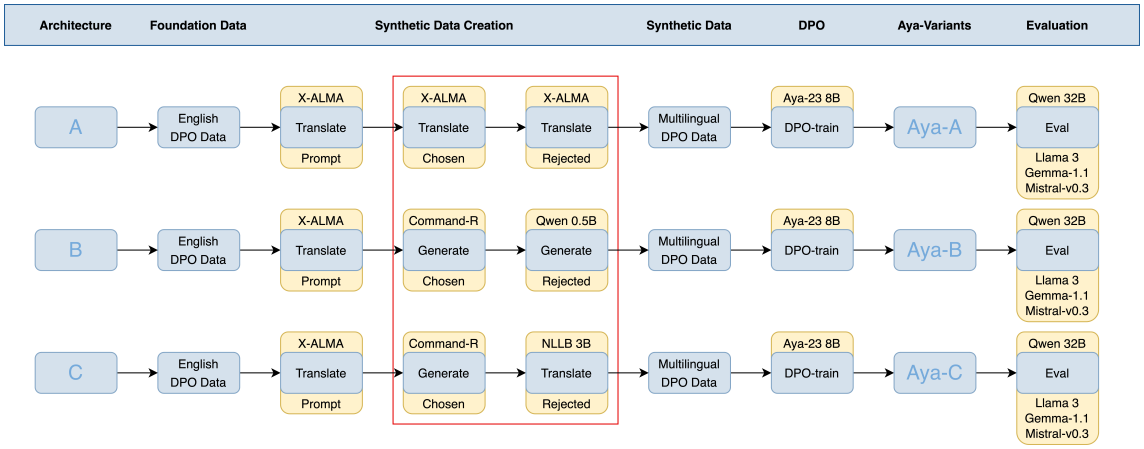

Figure 2: An overview of the experimental setup used in this paper. All architectures begin with the same custom English DPO dataset. We then synthetically expand this dataset to 9 languages, following three different architectural approaches. Next, we train an LLM with each dataset, resulting in the different Aya model variants. Finally, we assess the architectures' performance against three baselines. The red outline highlights the important differences in the architectures.

4.1 Assessing Translation Quality

4.1.1 WMT24++

We assess the translation quality of the models using the WMT24++ benchmark (Deutsch et al. 2025). The benchmark covers 55 languages, including our eight selected target languages. WMT24++ extends the WMT24 benchmark, which included only nine languages. All entries in the dataset have human references and post-edits. Since we are only interested in the performance of the tested models in our chosen target languages, we filter out the remaining 47 languages from the dataset.

4.1.2 Models

We assess the performance of eight different models. The selected models are a combination of widely used, computationally viable, and state-of-the-art models (Table 3). All models, except for the two GPT models, are run locally. For faster local inference, we use the vLLM (Kwon et al. 2023) Python library. Unfortunately, the library does not support running the X-ALMA model with beam search enabled. Therefore, we also include the GitHub variant of the model, which

allows for the use of beam search for decoding. For each model, we use the default chat templates and temperature settings as specified in their documentation (Table 3).

Canonical Name	Model Identifier	Size	Repo
Gemma-3	gemma-3-12b-it	12B	HuggingFace
GPT-3.5 Turbo	gpt-3.5-turbo-0125	$\approx$ 175B	OpenAI
GPT-4o	gpt-4o-2024-08-06	$\approx$ 1.8T	OpenAI
NLLB-200	nllb-200-3.3B	3.3B	HuggingFace
Qwen 2.5	Qwen2.5-14B-Instruct	14B	HuggingFace
TowerInstruct	TowerInstruct-13B-v0.1	13B	HuggingFace
X-ALMA	X-ALMA	29B	HuggingFace
X-ALMA-BS	X-ALMA w/ Beam Search	29B	GitHub

Table 3: The models used in this paper with identifiers, parameter size, and source links.

The initial WMT24++ dataset contains 1,000 entries for each language. However, we first filter out any entries labeled as bad sources. Second, we reduce the number of entries for each language to 300 to save computation. As we show in Appendix A.1, the performance hierarchy of the tested models becomes stable. Since we are not interested in the models' exact metric scores but only in selecting the best-performing model, it is empirically valid to downsample in this case.

4.1.3 Metrics

For the evaluation of the models, we generate responses for the WMT24++ dataset and assess their performance in the eight target languages using two metrics.

BLEU. The Bleu score measures the similarity between a generated response and the reference response. It does so by comparing the word n-grams of the generated response with those of the reference. Bleu calculates matches between the two texts for up to $n = 4$. Scores range between 0 and 1. Achieving a score of 1 is very rare, as it would indicate that the two translations are identical. Therefore, a good translation does not necessarily require a perfect score (Papineni et al. 2002).

CHRF++. The standard character n-gram F-score (chrF) works similarly to the Bleu score, with the main difference being that chrF compares n-grams at the character level instead of the word level. Since the standard chrF score can be overly optimistic in its evaluation, chrF++ incorporates word n-grams up to $n = 2$ to balance it. Scores range between 0 and 1. Again, achieving a perfect score of 1 is highly unlikely, as it would indicate that the translations are identical (Popović 2015, 2017; Post 2018).

4.2 Architecture A

As we show in Section 5.1, the best-performing open-source model is X-ALMA with beam search enabled. However, this model is computationally too expensive, and we are therefore forced to use the second-best open-source model: X-ALMA without beam search. We use this model for architecture A. This means that we translate the prompts, chosen responses, and rejected responses into the eight target languages using the X-ALMA model. We run the model locally using the vLLM library, with a maximum token length of 2,048 and a temperature of 0.6.

4.3 Architecture B

For this architecture, we use the same translated prompts as in architecture A. However, we then generate the chosen and rejected responses in the nine target languages (including English). For the chosen responses, we use the multilingual-optimized Cohere Command-R 35B model (Cohere 2024; Cohere and Labs 2024). For the rejected responses, we use the smaller but also multilingual-optimized Qwen2.5 0.5B Instruct model (Team 2024; Yang et al. 2024). Both models are run locally using the vLLM library, with a maximum token length of 2,048 and a temperature of 0.7.

4.4 Architecture C

For architecture C, we use the translated prompts from architecture A and the generated chosen responses from architecture B. For the rejected responses, we translate the English preference optimization dataset into the eight target languages using the worst-performing model identified in Section 5.1, which is Facebook's 3.3B-parameter variant of the NLLB-200 model (Meta AI 2022a,b). We once again run the model locally using the vLLM library, with a maximum token length of 2,048 and its default temperature.

4.5 Direct Preference Optimization

After creating datasets A, B, and C, we fine-tune the multilingual-optimized Aya-23-8B model (Aryabumi et al. 2024). For training the model using DPO, we use the OpenRLHF Python library (Hu et al. 2024). For each of the three datasets, we use the same hyperparameters for DPO, which are listed in Table 4.

Parameter	Value
Training Batch Size	128
Micro Batch Size	1
Max Sequence Length	4096
Epochs	1
Learning Rate	5e-6
Beta (DPO)	0.1
Zero Redundancy Optimization (Stage)	3

Table 4: The hyperparameters used for training the Aya-23-8B with DPO using the OpenRLHF library and our three created datasets.

4.6 Evaluation

For the remainder of this paper, we refer to the Aya-23-8B model trained on the dataset from architecture A as Aya-A, the model trained on the dataset from architecture B as Aya-B, and the model trained on the dataset from architecture C as Aya-C. The standard, untrained Aya-23-8B model is referred to as Aya-Base. To evaluate each of the Aya variants, we assess their win rates against three state-of-the-art models: (a) Meta Llama 3 8B Instruct (AI@Meta 2024), (b) Gemma 1.1 7B Instruct (Google 2024), and (c) Mistral 7B Instruct v0.3 (Mistral-AI 2024).

We use the Aya Evaluation Suite (Singh et al. 2024) as the dataset for generating responses from the Aya model variants. For each supported language, we replace the machine-translated data with the corresponding human-edited entries. We then sample 150 entries for each of our preference dataset languages, resulting in an evaluation dataset of 1,350 entries.

To assess the win rates of the Aya-Base and DPO variants, we use the Qwen2.5 32B Instruct model (Team 2024; Yang et al. 2024). The judge prompt used for evaluation is provided in Appendix B.1. To ensure stability, we report the average win rates over five runs. To mitigate positional bias when presenting the responses of models A and B, we randomize their order. The judge model is instructed to select its preferred response: A, B, or a Tie.

4.7 Ablations

To evaluate the impact of individual components of the pipeline, we make two ablations. In the first ablation, we construct our foundational English preference dataset using only 2 tasks: Chatbot Arena and HelpSteer2. To balance the number of entries, we increase the size of each of these datasets to 625, resulting in a final dataset size of 11,250. In the second ablation, we reduce the number of target languages to 4: Chinese, Dutch, Russian, and Spanish. To maintain the same overall dataset size, we increase the number of entries per task to 450, again yielding a final dataset size of 11,250.

Apart from the English foundational dataset, the rest of the architecture remains unchanged. We evaluate on all eight target languages plus English, which means that in the second ablation, we are also assessing the impact of the architecture on out-of-domain languages, thereby evaluating the model's zero-shot capabilities.

4.8 Computational Requirements

All experiments are conducted on the Snellius Cluster (the Dutch national supercomputer). To provide a reference for computational time and cost, we report the approximate runtime hours, system billing unit (SBU) costs, and types of GPUs used for various tasks in our experiments (Table 5). For SBU calculations, we follow the rule that an NVIDIA A100 40GB GPU takes 128 SBUs per hour, and an NVIDIA H100 80GB GPU 192 SBUs per hour.

Task	GPU	Cost (Time)	Cost (SBU)
X-ALMA 29B Translations	H100 40GB	≈ 18 hours	$\approx 3,456$ SBUs
NLLB-200 3.3B Translations	H100 80GB	≈ 8.5 hours	$\approx 1,632$ SBUs
Command-R 35B Generations	H100 80GB	≈ 5 hours	≈ 960 SBUs
Qwen2.5 0.5B Generations	A100 40GB	≈ 0.5 hours	≈ 64 SBUs
DPO fine-tuning	H100 80GB	≈ 1 hour	≈ 192 SBUs
Judge evaluation	H100 80GB	≈ 1 hour	≈ 192 SBUs
X-ALMA-BS 29B Translations (estimate)	4 H100 80GB	≈ 25 hours	$\approx 19,200$ SBUs

Table 5: Computational requirements for translation, generation and evaluation tasks.

5 Results

5.1 Assessing Translation Quality

In Figures 3 and 4, we analyze the BLEU and chrF scores for the tested models. The models are displayed in descending order based on their average scores across the eight target languages. An empty tile indicates that the model does not support the corresponding target language. To ensure that such models are not unfairly advantaged (for example, by not including low Arabic scores in their averages), we compute global averages for those models. Detailed averages for each model can be found in Appendix A.2, Table 11.

We observe that, for both metrics, the models generally follow the same performance trend. For instance, the best-performing model is GPT-4o, which is expected given that it is currently among the most advanced publicly accessible models. However, since it is proprietary and only available via a paid API, it is not a suitable choice for our experiments. The second-best performing model is X-ALMA with beam search enabled, run from the official GitHub codebase. In contrast, significantly smaller models, such as the NLLB-200 3.3B, perform the worst, particularly for Chinese and Japanese.

Another observation is that high-resource languages within LLMs and datasets, such as Chinese and Spanish, receive higher scores across the board. In contrast, more difficult languages, such as Arabic, tend to score significantly lower across all models; even GPT-4o struggles with them. Notably, Icelandic, despite being a very low-resource language, still achieves reasonable scores. This could possibly be due to its relation to higher-resource languages such as Dutch and German, sharing the same script, language family, and subgroup (see Table 2).

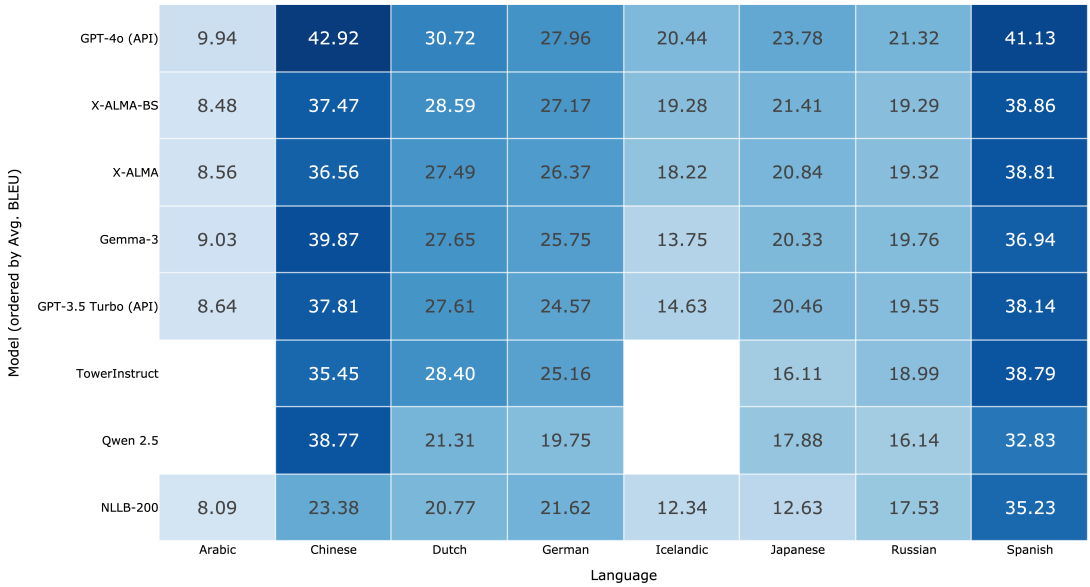

Figure 3: BLEU scores for all target languages and tested models. Empty cells indicate that the model does not support the target language. Models are ordered in descending order based on their average score across languages. Global averages are used when a language is unsupported by a model. The heatmap is based on Deutsch et al. (2025).

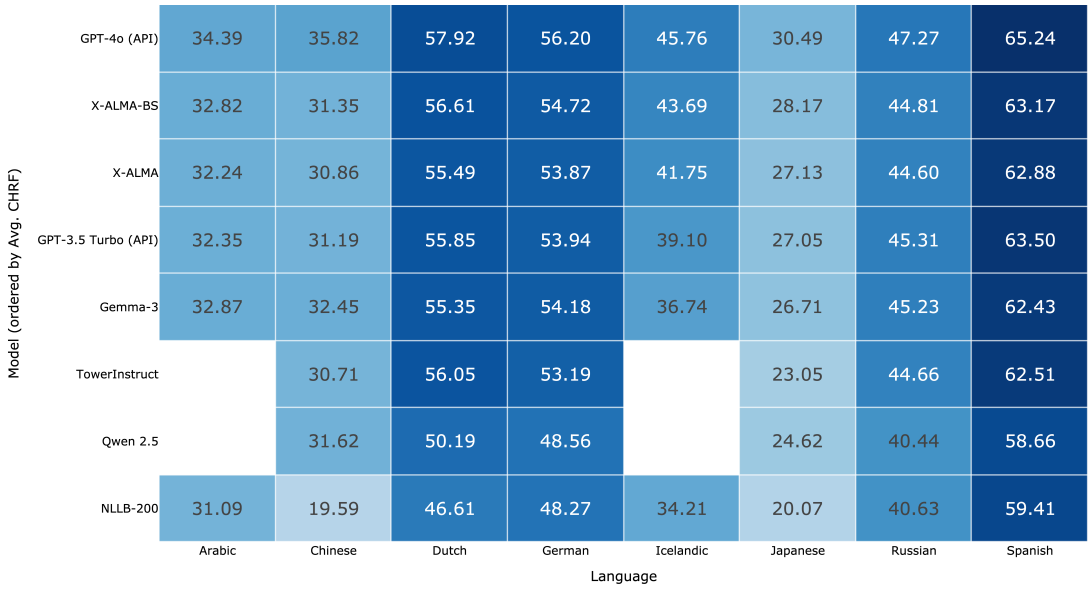

Figure 4: CHRF++ scores for all target languages and tested models. Empty cells indicate that the model does not support the target language. Models are ordered in descending order based on their average score across languages. Global averages are used when a language is unsupported by a model. The heatmap is based on Deutsch et al. (2025).

5.2 Direct Preference Optimization

In Table 6, we can observe the win rates for the Aya-Base, and Aya-A,B,C model variants. Here, the win percentage refers to the times the judge model preferred the Aya response over the baseline models, while the loss percentage refers to the times the judge model preferred the baseline response. As ties are not included in the table, the win-loss delta is most indicative of the model's performance. For readability, the standard deviation across the five runs is omitted from the table, but can be found in Appendix C.1.

The Aya-Base model already achieves strong win-loss against the three baseline models, with the Llama-3 model yielding the fewest losses against it. Aya-B improves performance over Aya-Base across all comparisons, with a gain of 5.69% against Llama-3, 3.43% against Gemma, and 6.14% against Mistral. Aya-C shows similar improvements, slightly outperforming Aya-B against the Gemma model by 1.10%, but underperforming against Mistral by 4.56%, and Llama-3 by 2.07%. Aya-A does not manage to improve the performance over the Aya-Base variant; it actually achieves lower win-loss rates than the baseline, with a drop of 4.32% against Llama-3, 4.15% against Gemma, and 2.27% against Mistral.

		Average 9 Languages		
		Win%	**Loss%**	**ΔW-L%**
Base	Llama-3	64.27%	35.27%	28.99%
	Gemma-1.1	69.56%	29.97%	39.59%
	Mistral-v0.3	69.10%	29.69%	39.41%
DPO A	Llama-3	62.16%	37.50%	24.67%
	Gemma-1.1	67.44%	32.00%	35.44%
	Mistral-v0.3	68.13%	30.99%	37.14%
DPO B	Llama-3	67.08%	32.40%	34.68%
	Gemma-1.1	71.29%	28.27%	43.02%
	Mistral-v0.3	72.32%	26.77%	45.55%
DPO C	Llama-3	66.15%	33.54%	32.61%
	Gemma-1.1	71.84%	27.70%	44.13%
	Mistral-v0.3	70.00%	29.01%	40.99%

Table 6: Win-loss for the Aya-Base model and DPO-optimized variants against the Meta Llama 3 8B Instruct, Gemma 1.1 7B Instruct, and Mistral 7B Instruct v0.3 models. Averages across 9 languages. Scores reported over 5 runs. Table based on Dang et al. (2024).

If we analyze the results per language of the Aya-Base model (Table 7), we can make some interesting observations. Aya-Base struggles with languages that it has not been specifically fine-tuned with, such as Icelandic, achieving win-loss rates far in the negatives against Gemma and Llama-3, and only just positive against Mistral. When looking at English, only against the Gemma model does the Aya-Base model achieve a high win-loss rate (41.07%), whereas it yields negative win-loss rates against both Llama-3 (-11.47%) and Mistral (-13.47%).

Language	**Llama 3 8B**			**Gemma 1.1 7B**			**Mistral v0.3**		
	Win%	Loss%	ΔW-L%	Win%	Loss%	ΔW-L%	Win%	Loss%	ΔW-L%
Arabic	85.33	14.67	70.67	79.87	20.13	59.73	93.60	6.40	87.20
Chinese	87.60	12.00	75.60	66.00	33.60	32.40	76.93	22.27	54.67
Dutch	56.93	43.07	13.87	72.40	27.60	44.80	65.47	34.53	30.93
English	43.60	55.07	-11.47	70.53	29.47	41.07	42.13	55.60	-13.47
German	68.13	31.87	36.27	76.27	23.73	52.53	72.67	27.33	45.33
Icelandic	20.13	78.53	-58.40	28.53	68.27	-39.73	46.67	45.87	0.80
Japanese	91.60	8.40	83.20	77.73	21.60	56.13	86.40	13.60	72.80
Russian	72.67	26.53	46.13	77.33	22.67	54.67	71.20	28.80	42.40
Spanish	52.40	47.33	5.07	77.33	22.67	54.67	66.80	32.80	34.00

Table 7: Per-language win/loss rates and ΔW-L of the Aya-Base against the Meta Llama 3 8B Instruct, Gemma 1.1 7B Instruct, and Mistral 7B Instruct v0.3 models. Scores are averaged over 5 runs.

These deficits in the Aya-Base model are improved when trained with DPO (Figure 5). Aya-B shows significant performance gains in English, outperforming Aya-Base against Gemma by 8.40%, Llama-3 by 15.74%, and Mistral by 13.87%. For Icelandic, the Aya-B variant also improves over Aya-Base, with a 6.53% gain against Llama-3 and an 8.27% gain against Mistral. Harder to explain is the performance drop for the Gemma model in Icelandic, where performance decreases by 9.6%. This drop is a major outlier in the general trend of performance improvements.

A final observation is that even for languages where the Aya-Base model already strongly outperforms the other models, the Aya-B variant still achieves further gains, such as increasing the win-loss rate for Russian against Gemma by 8.53% and for Japanese against Llama-3 by 4.00%. The in-depth per language win-loss rates for the Aya variants can be found in Appendix C.

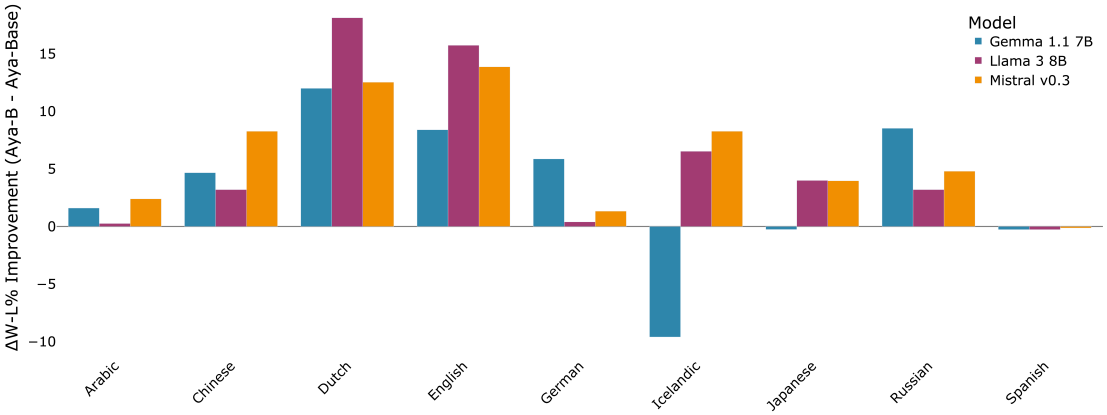

Figure 5: Performance gains of Aya-B over Aya-Base against the Meta Llama 3 8B Instruct, Gemma 1.1 7B Instruct, and Mistral 7B Instruct v0.3 models. Scores reported over 5 runs.

5.3 Ablations

5.3.1 Less Tasks

As stated in the experimental setup, the *less tasks* ablation removes three datasets from the foundational English data while keeping the original size intact.

		Average 9 Languages		
		Win%	**Loss%**	**ΔW-L%**
Base	Llama-3	64.27%	35.27%	28.99%
	Gemma-1.1	69.56%	29.97%	39.59%
	Mistral-v0.3	69.10%	29.69%	39.41%
DPO A	Llama-3	63.21%	36.22%	26.99%
	Gemma-1.1	69.04%	30.49%	38.55%
	Mistral-v0.3	68.80%	29.81%	38.99%
DPO B	Llama-3	67.47%	32.13%	35.33%
	Gemma-1.1	71.69%	27.94%	43.75%
	Mistral-v0.3	71.78%	26.89%	44.89%
DPO C	Llama-3	64.07%	35.38%	28.70%
	Gemma-1.1	69.57%	29.97%	39.60%
	Mistral-v0.3	69.08%	29.81%	39.27%

Table 8: Win-rates for the Aya-Base model and DPO-optimized variants against the Meta Llama 3 8B Instruct, Gemma 1.1 7B Instruct, and Mistral 7B Instruct v0.3 models. Averages across 9 languages. Scores reported over 5 runs. Table based on Dang et al. (2024).

Analyzing the results in Table 8, we see that Aya-A still achieves lower win-loss rates than the Aya-Base variant. Compared to using five tasks in the dataset, Aya-A now drops only 2% against Llama-3, 1.04% against Gemma, and 0.42% against Mistral. Aya-B achieves very similar results to using all tasks, with scores differing by only around 0.7% compared to the full dataset. Aya-C does not improve upon Aya-Base; its results differ by only around 0.3% from Aya-Base.

5.3.2 Less Languages

The *less languages* ablation removes four target languages, resulting in a synthetic dataset containing 5 languages (including English). Table 9 shows the performance of the Aya variants when evaluated across all 9 languages. Table 10 presents their performance when evaluated only on the languages not included during training, which are: German, Japanese, Icelandic, and Arabic.

When assessed across all 9 languages, we see results very similar to those observed when using all languages in the training process. Once again, Aya-A reduces performance compared to Aya-Base, while Aya-B and Aya-C improve upon Aya-Base. One noteworthy difference is that the performance gain of Aya-C over Aya-Base has increased to 6.22% against Mistral, compared to only 1.58% on the original dataset.

		Average 9 Languages		
		Win%	**Loss%**	**ΔW-L%**
Base	Llama-3	64.27%	35.27%	28.99%
	Gemma-1.1	69.56%	29.97%	39.59%
	Mistral-v0.3	69.10%	29.69%	39.41%
DPO A	Llama-3	62.89%	36.74%	26.15%
	Gemma-1.1	68.61%	30.79%	37.82%
	Mistral-v0.3	68.06%	30.90%	37.16%
DPO B	Llama-3	66.36%	32.80%	33.56%
	Gemma-1.1	70.74%	28.80%	41.94%
	Mistral-v0.3	72.03%	26.96%	45.07%
DPO C	Llama-3	67.50%	31.90%	35.60%
	Gemma-1.1	70.90%	28.59%	42.31%
	Mistral-v0.3	72.07%	26.44%	45.63%

Table 9: Win rates for the Aya-Base model and DPO-optimized variants against the Meta Llama 3 8B Instruct, Gemma 1.1 7B Instruct, and Mistral 7B Instruct v0.3 models. Averages across 9 languages. Scores reported over 5 runs. Table based on Dang et al. (2024).

In Table 10, we observe the zero-shot performance of the Aya variants. For all variants, the win-loss rates against Mistral increase, even for Aya-A. However, Aya-A shows a drop in performance compared to Aya-Base against both Llama and Gemma. Aya-B also exhibits a slight decrease in performance against Llama and Gemma. Aya-C demonstrates the best zero-shot capabilities, improving over Aya-Base with a 0.53% increase against Llama, a 2.52% increase against Gemma, and a 4.75% increase against Mistral.

		Average 4 Out-of-Domain Languages		
		Win%	**Loss%**	**ΔW-L%**
Base	Llama-3	66.80%	33.20%	33.60%
	Gemma-1.1	65.93%	34.08%	31.85%
	Mistral-v0.3	74.33%	25.38%	48.95%
DPO A	Llama-3	65.07%	34.60%	30.47%
	Gemma-1.1	64.80%	34.30%	30.50%
	Mistral-v0.3	75.27%	22.93%	52.34%
DPO B	Llama-3	65.90%	33.20%	32.70%
	Gemma-1.1	65.23%	33.87%	31.37%
	Mistral-v0.3	75.40%	22.53%	52.87%
DPO C	Llama-3	66.73%	32.60%	34.13%
	Gemma-1.1	66.87%	32.50%	34.37%
	Mistral-v0.3	76.17%	22.47%	53.70%

Table 10: Win rates for the Aya-Base model and DPO-optimized variants against the Meta Llama 3 8B Instruct, Gemma 1.1 7B Instruct, and Mistral 7B Instruct v0.3 models. Averages across 4 languages. Scores reported over 5 runs. Table based on Dang et al. (2024).

6 Discussion

6.1 DPO Averages

As discussed in Section 5.2, Aya-B and Aya-C result in slight performance improvements over the base Aya variant. Interestingly, Aya-A does not show this improvement. In fact, its performance against the three baselines is even worse. The main difference between the three architectures is that B and C rely on lower-quality negatives, rejected responses where the language quality is inferior to that of the chosen response. For both architectures B and C, this helps the model explicitly learn to avoid poor examples, thereby improving its performance. This is not the case for architecture A, which may instead steer the model toward other optimizations (such as helpfulness or question answering in the target languages), without resulting in better multilingual language output.

6.2 DPO per Language

As we saw with the per-language results, the optimized Aya-B variant improves performance over the Aya-Base model in almost every language. The only language that does not show improvement is Spanish. A possible reason for this could be noise in the architecture pipeline, which may particularly hinder Spanish due to its many dialects (e.g., Mexico, Spain, Colombia). However, since performance does not degrade either, this remains a quirk that could be addressed in future work.

More interesting is the significant performance loss in the specific case of Icelandic, where Aya-B's performance relative to Aya-Base drops against Gemma, an effect not seen against Llama-3 and Mistral. A possible hypothesis for the drop is that the Icelandic training data was particularly noisy, possibly introducing a specific flaw or error in the model's Icelandic capabilities. Although Aya-B still shows performance improvements, this specific flaw resulted in the judge model preferring the Gemma model more often compared to Aya-Base. Given the many moving parts in the pipeline and the lack of judge explanations, it is very difficult to pinpoint exactly what is causing this effect.

6.3 Ablations

Less Tasks. The *less tasks* ablation resulted in a slightly smaller drop in performance for Aya-A compared to Aya-Base. A possible reason for this could be that the extra tasks present in the full dataset introduce more noise when directly translated into the target languages. When these

are removed, particularly the Tasksource and Math-Step datasets, which are difficult to translate, the performance does not degrade as much. Comparing architectures B and C, we observed that Aya-C neither improves nor degrades performance relative to Aya-Base, whereas Aya-B produces results very similar to those obtained with the full dataset. This suggests that architecture C is more reliant on specific English preference data. For instance, the Gutenberg data, with its very long responses, may be crucial for architecture C.

Less Languages. When the models are assessed across all nine languages, we observe trends very similar to those found with the complete dataset. This suggests that the models may exhibit some degree of zero-shot capability for out-of-domain languages. However, as shown in Table 10, this effect is primarily seen with the Aya-C variant. Aya-A and Aya-B only outperformed Aya-Base when compared to the Mistral model. These somewhat inconsistent results highlight a limitation of our study, namely, the relatively small number of languages considered. Evaluating out-of-domain capabilities on just four languages may lead to inconsistent or unrepresentative results.

6.4 Architecture

In the related work section, we discussed the work of Dang et al. (2024), which corresponds to our architecture C. With architecture B, we present a competitive alternative method for improving the multilingual performance of LLMs using synthetic data. One could even argue that architecture B offers several advantages over C, the main one being that, instead of relying on noisy translation data for the rejected responses, it uses a lower-performing generative model, which allows for greater control over the pipeline. The less tasks ablation resulted in architecture C failing to improve the model's performance, possibly because a crucial task was removed from the data. In contrast, architecture B maintained its performance–demonstrating greater robustness.

6.5 Limitations

Although we find that two of the three architectures improve multilingual performance, there are some limitations to our work. The first is the dataset size. Even though we managed to increase the win-loss rates for the Aya-Base model, our dataset remains quite small; especially the foundational English base preference dataset.

Another limitation is that we created our own English preference dataset by combining five existing DPO datasets into one. This approach required several pre-processing steps, as previously discussed. Although we attempted to preserve the original quality of the data, this method is not perfect and may have introduced noise into the dataset. The same concern applies to our use of translation and generative models.

A final limitation is the lack of explanations in the judge evaluations. While we observe an increase in performance, we cannot provide a human-understandable explanation for why this improvement occurs. Future work could therefore focus on integrating an explainability framework into the judge evaluation process.

7 Conclusion

In this paper, we examined three architectures for creating synthetic multilingual preference data. We built all three from scratch using a custom English DPO preference optimization dataset. Each dataset was used to train a base model using DPO, and its performance was assessed against three different baselines. Our main finding is that architectures which generate the chosen responses in the target languages lead to improved multilingual performance. This is because such responses are more natural in the target languages and avoid English biases. When trained with Architecture B or C, Aya shows improved multilingual capabilities across both high-resource and low-resource languages, while also enhancing its performance in English.

References

AI@Meta (2024). "Llama 3 Model Card". In: URL: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

Aryabumi, Viraat et al. (2024). *Aya 23: Open Weight Releases to Further Multilingual Progress*. arXiv: 2405.15032 [cs.CL].

Bai, Yuntao, Andy Jones, et al. (2022). "Training a helpful and harmless assistant with reinforcement learning from human feedback". In: *arXiv preprint arXiv:2204.05862*.

Bai, Yuntao, Saurav Kadavath, et al. (2022). "Constitutional ai: Harmlessness from ai feedback". In: *arXiv preprint arXiv:2212.08073*.

Christiano, Paul F et al. (2017). "Deep reinforcement learning from human preferences". In: *Advances in neural information processing systems* 30.

Cohere (2024). *The Command R Model (Details and Application)*. URL: https://docs.cohere.com/docs/command-r.

Cohere and Cohere Labs (2024). *c4ai-command-r-v01*. DOI: 10.57967/hf/3139. URL: https://huggingface.co/CohereLabs/c4ai-command-r-v01.

Dang, John et al. (2024). "Rlhf can speak many languages: Unlocking multilingual preference optimization for llms". In: *arXiv preprint arXiv:2407.02552*.

Deutsch, Daniel et al. (2025). "Wmt24++: Expanding the language coverage of wmt24 to 55 languages & dialects". In: *arXiv preprint arXiv:2502.12404*.

Google (2024). *Gemma 1.1 7B IT Model Card*. Version 1.1. URL: https://huggingface.co/google/gemma-1.1-7b-it.

Hu, Jian et al. (2024). "OpenRLHF: An Easy-to-use, Scalable and High-performance RLHF Framework". In: *arXiv preprint arXiv:2405.11143*.

Jondurbin (2024). *Gutenberg-DPO v0.1*. https://huggingface.co/datasets/jondurbin/gutenberg-dpo-v0.1.

Kwon, Woosuk et al. (2023). "Efficient Memory Management for Large Language Model Serving with PagedAttention". In: *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Lai, Viet Dac et al. (2023). "Okapi: Instruction-tuned large language models in multiple languages with reinforcement learning from human feedback". In: *arXiv preprint arXiv:2307.16039*.

Meta AI (July 2022a). *200 languages within a single AI model: A breakthrough in high-quality machine translation*. Blog Post. URL: https://ai.meta.com/blog/nllb-200-languages-ai-model/.

— (2022b). *nllb-200-3.3B*. URL: https://huggingface.co/facebook/nllb-200-3.3B.

Mistral-AI (2024). *Model Card for Mistral-7B-Instruct-v0.3*. Version 0.3. URL: https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3.

Ouyang, Long et al. (2022). "Training language models to follow instructions with human feedback". In: *Advances in neural information processing systems* 35, pp. 27730–27744.

Pace, Alizée et al. (2024). *West-of-N: Synthetic Preferences for Self-Improving Reward Models*. arXiv: 2401.12086 [cs.CL]. URL: https://arxiv.org/abs/2401.12086.

Papineni, Kishore et al. (2002). "Bleu: a method for automatic evaluation of machine translation". In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318.

Popović, Maja (Sept. 2015). "chrF: character n-gram F-score for automatic MT evaluation". In: *Proceedings of the Tenth Workshop on Statistical Machine Translation*. Lisbon, Portugal: Association for Computational Linguistics, pp. 392–395. DOI: 10.18653/v1/W15-3049. URL: https://aclanthology.org/W15-3049.

— (Sept. 2017). "chrF++: words helping character n-grams". In: *Proceedings of the Second Conference on Machine Translation*. Copenhagen, Denmark: Association for Computational Linguistics, pp. 612–618. DOI: 10.18653/v1/W17-4770. URL: https://aclanthology.org/W17-4770.

Post, Matt (Oct. 2018). "A Call for Clarity in Reporting BLEU Scores". In: *Proceedings of the Third Conference on Machine Translation: Research Papers*. Belgium, Brussels: Association for Computational Linguistics, pp. 186–191. URL: https://www.aclweb.org/anthology/W18-6319.

Rafailov, Rafael et al. (2023). "Direct preference optimization: Your language model is secretly a reward model". In: *Advances in Neural Information Processing Systems* 36, pp. 53728–53741.

Schulman, John et al. (2017). "Proximal policy optimization algorithms". In: *arXiv preprint*. arXiv: 1707.06347 [cs.LG].

She, Shuaijie et al. (2024). "Mapo: Advancing multilingual reasoning through multilingual alignment-as-preference optimization". In: *arXiv preprint arXiv:2401.06838*.

Sileo, Damien (May 2024). "tasksource: A Large Collection of NLP tasks with a Structured Dataset Preprocessing Framework". In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. Ed. by Nicoletta Calzolari et al. Torino, Italia: ELRA and ICCL, pp. 15655–15684. URL: https://aclanthology.org/2024.lrec-main.1361.

Singh, Shivalika et al. (2024). *Aya Dataset: An Open-Access Collection for Multilingual Instruction Tuning*. arXiv: 2402.06619 [cs.CL].

Team, Qwen (Sept. 2024). *Qwen2.5: A Party of Foundation Models*. URL: https://qwenlm.github.io/blog/qwen2.5/.

Wang, Zhilin, Alexander Bukharin, et al. (2024). *HelpSteer2-Preference: Complementing Ratings with Preferences*. arXiv: 2410.01257 [cs.LG]. URL: https://arxiv.org/abs/2410.01257.

Wang, Zhilin, Yi Dong, et al. (2024). *HelpSteer2: Open-source dataset for training top-performing reward models*. arXiv: 2406.08673.

Xiao, Wenyi et al. (2024). "A Comprehensive Survey of Direct Preference Optimization: Datasets, Theories, Variants, and Applications". In: *arXiv preprint arXiv:2410.15595*.

Xinlai (2024). *Math-Step-DPO-10K*. https://huggingface.co/datasets/xinlai/Math-Step-DPO-10K.

Yang, An et al. (2024). "Qwen2 Technical Report". In: *arXiv preprint arXiv:2407.10671*.

Yuan, Weizhe et al. (2025). *Self-Rewarding Language Models*. arXiv: 2401.10020 [cs.CL]. URL: https://arxiv.org/abs/2401.10020.

Zheng, Lianmin et al. (2023). *Judging LLM-as-a-judge with MT-Bench and Chatbot Arena*. arXiv: 2306.05685 [cs.CL].

Zhou, Chunting et al. (2023). "Lima: Less is more for alignment". In: *Advances in Neural Information Processing Systems* 36, pp. 55006–55021.

A WMT24++

A.1 Downsampling

As we can observe in Figure 6 and 7, the hierarchy in performance of the models stabilizes as we increase the sample size. This consequently means that in order to save the heavy compute (especially for the X-ALMA-BS model) assessing the models' performance on a lower sample size of the WMT24++ dataset does not change the outcome if all of the data were used.

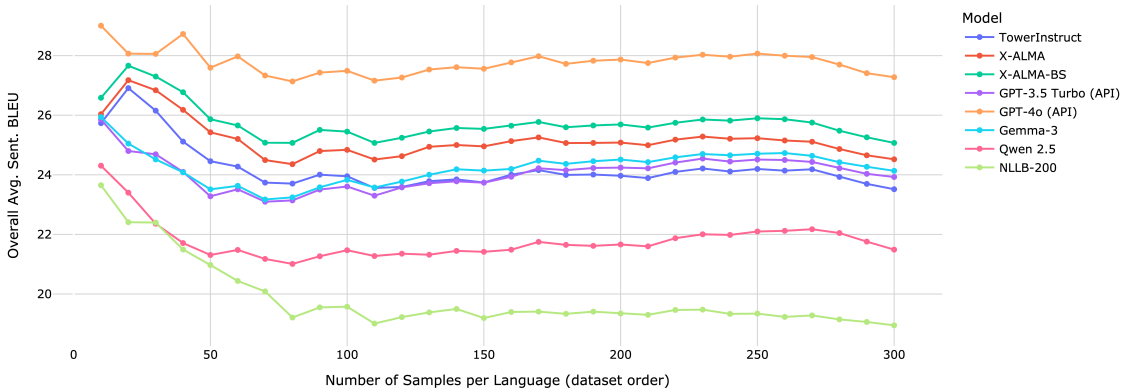

Figure 6: BLEU score average of the WMT24++ benchmark dataset with different sample sizes. Global averages are used if a language is unsupported by a model. Sample sizes taken from 0 to 300 in steps of 10. Y-axis shows the average BLEU scores. X-axis shows the number of samples. All tested models.

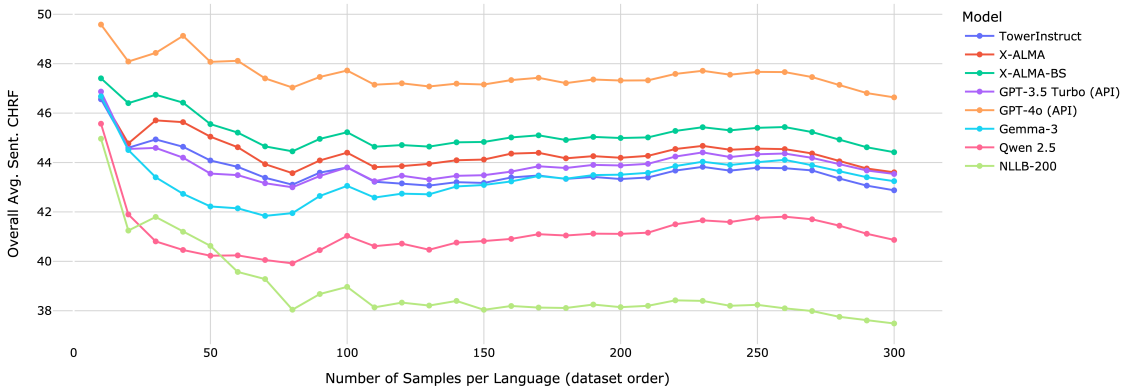

Figure 7: CHRF++ score average of the WMT24++ benchmark dataset with different sample sizes. Global averages are used if a language is unsupported by a model. Sample sizes taken from 0 to 300 in steps of 10. Y-axis shows the average CHRF++ scores. X-axis shows the number of samples. All tested models.

A.2 Model Summaries

Table 11 provides the in-depth averages scores for all tested models with their supported languages.

Canonical Name	BLEU (Avg)	CHRF++ (Avg)	Langs Cov.	Samples
GPT-4o	27.28	46.64	8/8	2400
X-ALMA-BS	25.07	44.42	8/8	2400
X-ALMA	24.52	43.60	8/8	2400
Gemma-3	24.13	43.25	8/8	2400
GPT-3.5 Turbo	23.93	43.54	8/8	2400
TowerInstruct	23.52	42.88	6/8	1800
Qwen 2.5	21.49	40.87	6/8	1800
NLLB-200	18.95	37.49	8/8	2400

Table 11: Model performance summary. Averages are calculated over all 8 languages. Global averages are used if a language is unsupported by a model.

B Evaluation

B.1 Judge Prompt

The judge prompt used for the Qwen2.5 32B model. The judge prompt is adapted from Dang et al. (2024).

System Message

You are a helpful assistant whose goal is to select the preferred (least wrong) output for a given instruction in {`language_name`}.

User Message

Which of the following answers is the best one for the given instruction in {`language_name`}? A good answer should follow these rules:

1. It should be in {`language_name`}.
2. It should answer the request in the instruction.
3. It should be factually and semantically comprehensible.
4. It should be grammatically correct and fluent.

Instruction: {`original_prompt`}
Answer (A): {`response_a`}
Answer (B): {`response_b`}

State only `A` or `B` to indicate your choice.
If both answers are equally good or bad, state `TIE`.
Your response should use the format:

```
Preferred:  <'A' or 'B' or 'TIE'>
```

C Direct Preference Optimization

C.1 Results with Standard Deviation

		Average 9 Languages		
		Win%	**Loss%**	**ΔW-L%**
Base	Llama-3	64.27 ± 0.59%	35.27 ± 0.49%	28.99 ± 1.08%
	Gemma-1.1	69.56 ± 0.65%	29.97 ± 0.67%	39.59 ± 1.31%
	Mistral-v0.3	69.10 ± 0.29%	29.69 ± 0.27%	39.41 ± 0.55%
DPO A	Llama-3	62.16 ± 1.01%	37.50 ± 1.08%	24.67 ± 2.08%
	Gemma-1.1	67.44 ± 0.27%	32.00 ± 0.37%	35.44 ± 0.63%
	Mistral-v0.3	68.13 ± 0.50%	30.99 ± 0.55%	37.14 ± 1.05%
DPO B	Llama-3	67.08 ± 0.60%	32.40 ± 0.47%	34.68 ± 1.06%
	Gemma-1.1	71.29 ± 0.39%	28.27 ± 0.40%	43.02 ± 0.80%
	Mistral-v0.3	72.32 ± 0.22%	26.77 ± 0.22%	45.55 ± 0.42%
DPO C	Llama-3	66.15 ± 0.27%	33.54 ± 0.44%	32.61 ± 0.71%
	Gemma-1.1	71.84 ± 0.81%	27.70 ± 0.77%	44.13 ± 1.58%
	Mistral-v0.3	70.00 ± 0.78%	29.01 ± 0.57%	40.99 ± 1.35%

Table 12: Win-rates for the Aya-Base model and DPO-optimized variants against the Meta Llama 3 8B Instruct, Gemma 1.1 7B Instruct, and Mistral 7B Instruct v0.3 models. Averages across 9 languages. Scores reported over 5 runs for stability. Table based on Dang et al. (2024).

C.2 Per-Language Win-Loss Rates for Aya-A

Language	**Llama 3 8B**			**Gemma 1.1 7B**			**Mistral v0.3**		
	Win%	Loss%	ΔW-L%	Win%	Loss%	ΔW-L%	Win%	Loss%	ΔW-L%
Arabic	84.00	16.00	68.00	80.27	19.73	60.53	94.40	5.60	88.80
Chinese	88.00	12.00	76.00	66.53	33.47	33.07	81.73	18.27	63.47
Dutch	59.20	40.80	18.40	75.07	24.53	50.53	66.53	33.47	33.07
English	41.73	56.40	-14.67	67.33	31.87	35.47	41.47	57.07	-15.60
German	62.80	36.67	26.13	74.40	25.60	48.80	71.73	28.00	43.73
Icelandic	17.20	82.53	-65.33	20.27	76.53	-56.27	41.07	53.47	-12.40
Japanese	88.93	11.07	77.87	73.07	26.27	46.80	86.67	13.33	73.33
Russian	68.67	30.93	37.73	75.33	24.67	50.67	67.33	32.00	35.33
Spanish	48.93	51.07	-2.13	74.67	25.33	49.33	62.27	37.73	24.53

Table 13: Per-language win/loss rates and ΔW-L performance of Aya-A against the Meta Llama 3 8B Instruct, Gemma 1.1 7B Instruct, and Mistral 7B Instruct v0.3 models. Scores are averaged over 5 runs.

C.3 Per-Language Win-Loss Rates for Aya-B

Language	**Llama 3 8B**			**Gemma 1.1 7B**			**Mistral v0.3**		
	Win%	Loss%	ΔW-L%	Win%	Loss%	ΔW-L%	Win%	Loss%	ΔW-L%
Arabic	85.47	14.53	70.93	80.67	19.33	61.33	94.80	5.20	89.60
Chinese	89.20	10.40	78.80	68.53	31.47	37.07	81.47	18.53	62.93
Dutch	66.00	34.00	32.00	78.40	21.60	56.80	71.73	28.27	43.47
English	51.73	47.47	4.27	74.27	24.80	49.47	50.00	49.60	0.40
German	68.00	31.33	36.67	79.20	20.80	58.40	73.07	26.40	46.67
Icelandic	23.07	74.93	-51.87	24.27	73.60	-49.33	51.60	42.53	9.07
Japanese	93.60	6.40	87.20	77.47	21.60	55.87	88.39	11.61	76.77
Russian	74.27	24.93	49.33	81.60	18.40	63.20	73.07	25.87	47.20
Spanish	52.40	47.60	4.80	77.20	22.80	54.40	66.80	32.93	33.87

Table 14: Per-language win/loss rates and ΔW-L performance of Aya-B against the Meta Llama 3 8B Instruct, Gemma 1.1 7B Instruct, and Mistral 7B Instruct v0.3 models. Scores are averaged over 5 runs.

C.4 Per-Language Win-Loss Rates for Aya-C

Language	Llama 3 8B			Gemma 1.1 7B			Mistral v0.3		
	Win%	Loss%	ΔW-L%	Win%	Loss%	ΔW-L%	Win%	Loss%	ΔW-L%
Arabic	88.53	11.47	77.07	84.27	15.07	69.20	94.80	5.20	89.60
Chinese	91.60	8.40	83.20	71.33	28.40	42.93	84.27	15.73	68.53
Dutch	63.87	35.47	28.40	80.93	18.67	62.27	70.40	29.33	41.07
English	51.20	47.47	3.73	72.67	26.93	45.73	49.87	48.13	1.73
German	63.20	36.80	26.40	79.73	20.27	59.47	70.53	28.53	42.00
Icelandic	17.73	81.60	-63.87	19.47	78.13	-58.67	35.33	59.47	-24.13
Japanese	94.27	5.73	88.53	82.40	17.60	64.80	90.67	9.33	81.33
Russian	70.80	29.07	41.73	80.00	20.00	60.00	68.67	31.33	37.33
Spanish	54.13	45.87	8.27	75.73	24.27	51.47	65.47	34.00	31.47

Table 15: Per-language win/loss rates and ΔW-L performance of Aya-C against the Meta Llama 3 8B Instruct, Gemma 1.1 7B Instruct, and Mistral 7B Instruct v0.3 models. Scores are averaged over 5 runs.