Multilingual Reasoning via Self-training

Leonardo Ranaldi **Giulia Pucci**
School of Informatics, University of Edinburgh, UK
Department of Computing Science, University of Aberdeen, UK
{first_name.last_name}@ed.ac.uk

Abstract

Although reasoning is innately language-agnostic, the multilingual capacities remain a significant challenge for large language models (LLMs). Their ability to generate structured, step-wise explanations is constantly restricted to dominant languages in pre-training data, making cross-lingual generalisation difficult and hindering broader global adoption. Recent works have introduced eclectic strategies to enhance reasoning beyond English; however, they remain related to spoken language, which is not always optimal for reasoning.

To make LLMs' multilingual reasoning capabilities aligned and grounded, we propose a modular approach that instructs the models to structure reasoning in an abstractive problem space and then delivering step-wise reasoning trajectories, employing Self-training. Experiments show that our approach stably achieves significant improvements in the multilingual reasoning of various models and tasks, with improved reasoning consistency across languages.

1 Introduction

Reasoning is innately language-agnostic; indeed, multiple evidences show that human language is optimised for communication rather than reasoning (Mahowald et al., 2024). Yet, some complex reasoning requires step-wise passages to use extralinguistic constructions, such as formulas and symbols, as a tool for structuring and externalising thought processes (Amalric and Dehaene, 2016). However, in the era of large language models (LLMs), approaches such as Chain-of-Thought (CoT) *et inter alia* aim to emulate human reasoning and thought via language generation, which should not be conditioned by the spoken languages. Nevertheless, several works demonstrate that, due to the disparity in pre-training data across languages, the LLMs' reasoning capability differs between languages and dominant languages, such as English, which is much more performant than the others.

Research advances in multilingual reasoning are increasingly aimed at completing the performance differences among languages, enhancing the models' capabilities through in-context learning interventions (Huang et al., 2023; Ranaldi et al., 2024b; Li et al., 2024), supervised fine-tuning (SFT) strategies that differ from language-specific augmentation (Ranaldi and Pucci, 2023; Üstün et al., 2024) to task-oriented tuning (Zhang et al., 2024), preference optimisation (Dang et al., 2024) and encoder-based strategies (Yoon et al., 2024). Although these approaches enabled the production of performant solutions for transferring and aligning multilingual reasoning capabilities, we argue that some crucial challenges remain hindering further progress. Firstly, the actual benefits of in-context interventions are limited to large-scale LLMs, which are able to follow provided instructions systematically, but at the same time, they must have robust multilingual proficiency; therefore, many works rely on SFT techniques that maintain reduced costs when used to specialised smaller-scale LLMs. Secondly, they require huge amounts of complex reasoning annotations and tremendous tuning efforts to get multilingual LLMs capable of delivering reasoning through SFT and preference optimisation techniques. Finally, encoder-based approaches are a convenient solution for aligning multilingual representations. Still, they map the input into a latent space represented in symbols by limiting it to a single deterministic decoding path.

To improve LLMs' multilingual reasoning capabilities, we propose a modular approach that first instructs the models to formalise the problem abstractly and then deliver step-wise reasoning trajectories that lead to the reasoning in the same way for all languages. Our approach breaks down problem solutions into a sequence of formal, language-agnostic sub-problems, which are solved sequentially and can be more effectively utilised by models. The decomposition consists of two high-level

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 11566–11582

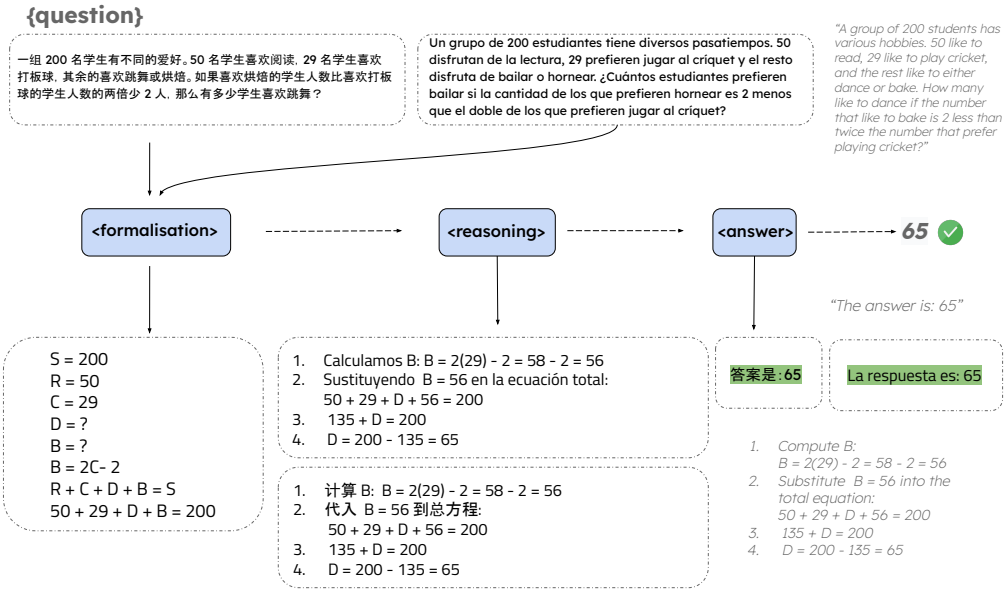

Figure 1: We enable models to deliver language-agnostic reasoning trajectories across languages by disentangling content from logical reasoning through structured abstraction operating via our Step-wise Abstractive Thought.

modules: *Formalisation* and *Reasoning Execution*. As shown in Figure 1, we instruct the models to: *(i)* identify the relevant information in the given problems, formalising the variable and predicates and delivering symbolic transformations; *(ii)* generate a reasoning execution where the transformations are solved using symbolic representations that explicitly exemplify the final solution, producing an answer where a final solution is delivered in the same query language.

Previous works proposed English-based strategies that operate via logical formalisms coupled with external symbolic solvers (Gaur and Saunshi, 2023; Pan et al., 2023). However, entirely symbolic approaches have the bottleneck of requiring a complete translation from natural to formal languages, which may negatively impact efficiency and flexibility, and add further obstacles.

To achieve a better trade-off, we treat formalisations in an eclectic manner and propose methods to disentangle content from logical reasoning without introducing rigorous formalisms. To this end, we instruct larger LLMs to generate synthetic demonstrations using Step-wise Abstractive Thought (SWATH); then we use the generated demonstrations for *Self-training* smaller LLMs. As classical tuning strategies behind a standard warm-up phase, we propose different alignment methods ranging from supervised fine-tuning (instruction-tuning) to preference optimisation techniques (Reinforcement Learning).

We performed an extensive empirical evaluation by observing the impact of different tuning and alignment methods. In multilingual reasoning tasks, our approaches demonstrated significant improvements, resulting in an overall increase in exact matching in proposed tasks, which led to the following results and conclusions:

- Structuring the LLMs' multilingual reasoning passages in formal symbolic trajectories (SWATH), which employ language-agnostic formalism, enhances accuracy and delivers more verifiable generations through a transparent and structured reasoning process.
- Operating with Self-training heuristics consisting of both tuning and preference optimisation enables robust, pervasive and language-aligned models. Indeed, heuristics based on tuning via synthetic demonstrations work well but do not deliver robust performant models in all languages; instead, heuristics based purely on preference optimisation strategies require onerous computational costs to achieve actual benefit.
- Our approach allows the disentanglement of content from logical reasoning, improving multilingual reasoning in LLMs, thus benefiting in different language spaces (we demonstrate this by proposing MGSM-SYMBOLIC).

2 Background

2.1 Improving Reasoning in Large Language Models

Improving reasoning capabilities in LLMs (both English and multi and cross-lingual) is usually conducted through SFT using ground-thought examples and preference-based approaches.

Supervised Fine-Tuning SFT is a common way to improve a model $\mathcal{M}$ towards a reasoning problem with the labelled dataset $\mathcal{L}$. Where $\mathcal{L}$ comprises a set of questions x, the corresponding y is a step-wise rationale leading to the answer a. The answers are extracted from the step-wise explanations using regular expressions. A delivered step-wise explanation $\hat{y}$ is valid if the extracted answer $\hat{a}$ matches the gold answer a. Formally, the labelled dataset with n instances can be represented as:

$$\mathcal{L} = \{(x^i, y^i, a^i)\}_{i=1}^{n}. \tag{1}$$

Operating via $\mathcal{L}$, SFT optimise $\mathcal{M}_\theta$ by minimising the negative log-likelihood loss:

$$\mathcal{L}_{\text{SFT}}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{L}}\left[\sum_{t=1}^{T} \log f_\theta(y_t|x, y_{1:t-1})\right], \tag{2}$$

where T is the length of the rationale y and we use y_t to represent the t-th token in y.

Self-training Self-training is a set of SFT strategies that have recently regained overall attention as a functional strategy (DeepSeek-AI et al., 2025). These approaches, usually, operate in a two-phase process: first, a base model $\mathcal{M}_\theta$ is trained on a subset of dataset $\mathcal{L}$, serving as teacher $\mathcal{M}_{\theta'}$, and is then used to generate a pseudo-labeled dataset $\hat{\mathcal{L}}$ by annotating an unlabeled dataset $\mathcal{L}$. In the second phase, a student model $\mathcal{M}_\theta$ is trained on a combination of $\mathcal{L}$ and $\hat{\mathcal{L}}$ to outperform the the teacher model $\mathcal{M}_{\theta'}$. Several works demonstrated that pseudo-label quality significantly influences the overall performance. Hence, Wang et al. (2024), propose a self-training that iteratively refine $\mathcal{M}_\theta$, ensuring the generation of higher-quality pseudo-labeled data at each step.

Reinforcement Learning Heuristics (RL) Within the Self-training approaches, Reinforcement Learning from Human Feedback (RLHF) is widely used for aligning language models with human feedback (Ouyang et al., 2022). The RLHF framework refines LLM behaviour by leveraging human preference data to guide model tuning through RL. Specifically, it uses a reward model $r(x, y)$, which captures human preference preferences given an input x and its corresponding output y. This reward model is then employed to assign preference scores to arbitrary LLM-generated outputs, facilitating iterative policy refinements via proximal policy optimisation (PPO) (Schulman et al., 2017). The training process follows an optimisation function, for instance, PPO, which optimises the model policy ϕ_θ to maximise expected rewards while minimising divergence from the SFT policy:

$$\mathbb{E}_{(x,y)\sim D_\pi}[r(x, y) - \gamma \log \frac{\phi_\theta(y|x)}{\phi_{\text{SFT}}(y|x)}], \tag{3}$$

where ϕ_{SFT} denotes the original model trained via SFT, and γ serves as a regularization hyperparameter to constrain policy updates.

Direct Preference Optimization Although RLHF via PPO effectively aligns models with diverse human preferences, it requires four sub-models, making training computationally expensive and complex. To avoid explicit reward model training, Rafailov et al. (2024) introduced Direct Preference Optimization (DPO), enabling direct tuning with human preference. This approach involves a warm-up phase performed (via SFT-style), then, given some novel instances x, are sampled completions from the policy model ϕ_{ref}:

$$y_1, y_2 \sim \phi_{\text{ref}}(\cdot \mid x). \tag{4}$$

Then, is builded the DPO dataset $\mathcal{L}_{DPO}$ from the completions based on human preference:

$$\mathcal{L}_{DPO} = \{(\, x^i,\, y_w^i,\, y_l^i\,)\}_{i=1}^{N}, \tag{5}$$

where y_w^i and y_l^i represent the winning and losing completions respectively. Then, $\mathcal{M}_\theta$ is optimised to minimise $\mathcal{L}_{\text{DPO}}(\phi_\theta; \phi_{\text{ref}})$ which can be defined as follows:

$$\mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}}\left[-\log\sigma\Big(r(y_w|x) - r(y_l|x)\Big)\right], \tag{6}$$

where $r(\cdot|x) = \beta \log \frac{\phi_\theta(\cdot|x)}{\phi_{\text{ref}}(\cdot|x)}$ and β is a coefficient that controls ϕ_θ's deviation from ϕ_{ref}.

DPO simplifies preference-based learning by eliminating explicit reward models, and it suffers

from limited generalisation across diverse tasks (Lin et al., 2024) since it relies on static pairwise comparisons of preference rankings and does not fully capture context-dependent variations.

Group Relative Policy Optimization To address these challenges, Shao et al. (2024a) introduced Group Relative Policy Optimization (GRPO), an extension of PPO designed to stabilise training by leveraging group-based reward estimation. GRPO constructs response groups and evaluates preferences relatively within each group. Given a batch of completions from the policy model ϕ_θ, GRPO assigns group-relative advantages rather than absolute preference scores, ensuring more stable updates across diverse tasks and languages. GRPO follows a loss function similar to PPO but replaces absolute reward estimation with relative ranking within groups:

$$\mathbb{E}_{(x,y)\sim D}\left[A_{\text{rel}}(y|x)\log\pi_\theta(y|x) - \beta D_{\text{KL}}\left(\pi_\theta\|\pi_{\text{ref}}\right)\right] \tag{7}$$

where: $A_{\text{rel}}(y|x)$ represents the relative advantage of the completion y compared to other completions within the same group, π_θ denotes the updated policy, π_{ref} represents the pre-trained policy before GRPO updates. D_{KL} ensures the updated policy remains close to its prior distribution, preventing unnecessary deviations. Finally, as in the previous cases, β is a hyperparameter controlling the strength of the KL penalty.

The relative advantage $A_{\text{rel}}(y|x)$ is computed as follows:

$$A_{\text{rel}}(y|x) = \frac{r(y|x) - \mu}{\sigma} \tag{8}$$

where $r(y|x)$ is the reward assigned to the response y, μ is the mean reward within the group and σ is the standard deviation of the group's reward distribution. GRPO has proven highly effective in multi-task reasoning, where grouping similar tasks allows for more stable optimization and better generalization across related problem structures. By leveraging relative comparisons within task groups, GRPO enables adaptive learning mechanisms that move beyond static absolute preference scores, ensuring more consistent and robust policy updates.

2.2 Multilingual Reasoning

Ongoing strategies assess the effectiveness of large language models (LLMs) by evaluating their ability to handle complex reasoning tasks, such as mathematical problem-solving. For these reasons, several mathematical reasoning tasks have been introduced to benchmark these capabilities, including GSM8K and SVAMP. To extend this evaluation to multilingual contexts, Shi et al. (2022) extent GSM8K beyond English (i.e., MGSM) by manually translating 250 samples from the GSM8K test set into multiple languages. Following this trend, Chen et al. (2023) introduced MSVAMP, which, in a similar way to Shi et al. (2022), provides SVAMP in different languages. Relying on these two multilingual evaluation tasks, different works proposed translation-based (Ranaldi et al., 2024a), supervised fine-tuning (SFT) (Üstün et al., 2024; Ghazaryan et al., 2025), and preference-based alignment (Dang et al., 2024) strategies yielding significant improvements in multilingual reasoning.

2.3 Motivations

The effectiveness of these approaches depends heavily on the quantity and quality of the training data. Hence, each of these methods has its weaknesses. While SFT is prone to challenges such as catastrophic forgetting and limited generalisation to out-of-domain problems, traditional alignment methods (introduced in §2.1) require Critic-based control systems, which may incur additional costs.

A final unexplored solution could be operating through sequential instructions, moving the effort into finding the most performant prompt. However, these use English as the language for reasoning (Huang et al., 2023), are suitable for particular tasks (Ranaldi et al., 2024b), and are ineffective for small LLMs (Ranaldi et al., 2024c). Finally, the nature of the reasoning is induced by the instructions provided by relating them to the constraints of the prompt compositions.

Our Proposal Reasoning is language-agnostic; however, the reasoning capabilities of LLMs differ between languages. We aim to disentangle content from logical reasoning by operating via language-agnostic formalisation that allows for aligning reasoning in languages beyond English. In other words, we transfer the problem from any language into a standard formalism that is easier to manipulate and common in all languages. We then conduct the reasoning phase and provide the answer in a specific language. To conduct these two phases, we instruct LLMs to follow this process and tune smaller models using Self-training strategies.

3 Method

To enable models to do this without the constraints of sequential instructions, we propose a Self-training approach that, in addition to the loss functions of the fine-tuning, employs different preference optimisation policies §3.1 to self-improve the models. We iteratively apply preference optimisation algorithms (RL) and fine-tuning (SFT), training the model to abstract the problem and deliver a step-wise formal solution (§3.2). The iterative process ends when the model performance converges or reaches the maximum iteration. A formal description of the proposed method is illustrated in Algorithm 1.

3.1 Preference Estimation

RL strategies employ preference estimation. This generally involves aligning the policy model with preferences using a reward model, which learns to predict preferences based on comparisons and leads the optimisation process. Although this approach is practical, it has problems with generalisation, scalability, robustness, and alignment. In DPO and GRPO, rule-based reward models are used. While DPO is generally based on a series of naive string-matching functions with ground truth values, rules are explicitly defined in GRPO. Accordingly, we define the following preference policies:

DPO Preference Estimation We use a string matching function to follow related approaches for English (Ranaldi and Freitas, 2024b; Wang et al., 2024). Moreover, we improve this by filtering out generations that follow the defined structure and well-defined form (described in Appendix N)

GRPO Preference Estimation Following Shao et al. (2024a) we define a set of rule-based metrics that control the accuracy, the structure and the form of the generations (described in Appendix M).

3.2 Self-training

Classic self-training begins with fine-tuning the base model $\mathcal{M}_\theta$ to optimise $\mathcal{D}_{\text{SFT}}$, resulting in an updated model $\mathcal{M}_{\theta'}$. Behind this stage, we assume that $\mathcal{M}_{\theta'}$ can solve a certain problem. Specifically, given a question x, $\mathcal{M}_{\theta'}$ delivers a formal statements $\hat{y}$ along with the corresponding answer $\hat{a}$.

Self-training In this phase, we first sample multiple outcomes $\hat{y}$ from $\mathcal{M}_{\theta'}$ for a set of questions x from $\mathcal{U}$. We apply preference estimation heuristics to build completions based on different policies (for DPO, we use couples, while for GRPO, we use groups of completions). These generations form the dataset $\mathcal{D}$, which is used to train the model with the objective functions ($\mathcal{L}_{\text{DPO}}$ and $\mathcal{L}_{\text{GRPO}}$), yielding an updated model $\mathcal{M}_{\theta^d}$.

Then we use $\mathcal{M}_{\theta^d}$ to generate a new pseudo-labeled dataset for the next-round tuning:

$$\mathcal{S} = (x, \hat{y}) | x \sim \mathcal{U}, \hat{y} \sim_\theta (\cdot|x). \tag{9}$$

After generation, the dataset $\mathcal{S}$ is refined by removing incorrect answers and eliminating duplicates. Consequently, the resulting pseudo-labeled dataset, denoted as $\mathcal{S}^\alpha$, is a subset of the original dataset, i.e., $\mathcal{S}^\alpha \subset \mathcal{S}$. The final training dataset is constructed by combining the original labeled dataset $\mathcal{L}$ with the newly generated pseudo-labeled dataset $\mathcal{S}^\alpha$. During this process, each new dataset is used to train from the original base model $\mathcal{M}_\theta$, rather than continually fine-tuning $\mathcal{M}_\theta$, to mitigate the risk of overfitting.

3.3 Single-training

For comparative purposes, we conduct individual training operating only with SFT, DPO and GRPO.

Algorithm 1 Self-training via DPO or GRPO

Input: pre-trained language model $\mathcal{M}_\theta$
labeled dataset $\mathcal{L} = \{(x^i, y^i, a^i)\}_{i=1}^{l}$
unlabeled dataset $\mathcal{U} = \{(x^i, a^i)\}_{i=1}^{u}$
mode $\in$ {DPO, GRPO}
Output: fine-tuned model $\mathcal{M}_{\theta'}$

Warm-up stage
Fine-tune $\mathcal{M}_\theta$ on $\mathcal{L}$ to get $\mathcal{M}_{\theta'}$
repeat
  if mode = DPO **then**
    Generate DPO dataset $\mathcal{D}$:
    $\mathcal{D} = \{(x^i, y_w^i, y_l^i)\}_{i=1}^{N}$
    where $x^i \sim \mathcal{U}$ and $y_w^i, y_l^i \sim \mathcal{M}_{\theta'}(\cdot|x^i)$
    Tune $\mathcal{M}_{\theta'}$ with $\mathcal{L}_{\text{DPO}}$ on $\mathcal{D}$ to get $\mathcal{M}_{\theta^d}$
  end if
  if mode = GRPO **then**
    Generate GRPO dataset $\mathcal{G}$:
    $\mathcal{G} = \{(x^i, G^i)\}_{i=1}^{N}$
    where $x^i \sim \mathcal{U}$
    and $G^i = \{y_1, \ldots, y_k\} \sim \mathcal{M}_{\theta'}(\cdot|x^i)$
    Compute relative preferences within each group G^i,
    assign pairwise relative scores to outputs in G^i.
    Tune $\mathcal{M}_{\theta'}$ with $\mathcal{L}_{\text{GRPO}}$ on $\mathcal{G}$ to get $\mathcal{M}_{\theta^g}$
  end if
  # SFT step
  Build pseudo-labeled dataset $\mathcal{S}$:
  $\mathcal{S} = \{(x^i, \hat{y}^i, \hat{a}^i)\}_{i=1}^{s}$
  where $x^i \sim \mathcal{U}$ and $\hat{y}^i, \hat{a}^i \sim \mathcal{M}_{\theta^d}(\cdot|x^i)$
  $\mathcal{M}_{\theta^g}(\cdot|x^i)$
  Select $\mathcal{S}^\alpha \subset \mathcal{S}$ when $\hat{a}^i = a^i$
  Update $\mathcal{L} \leftarrow \mathcal{S}^\alpha \cup \mathcal{L}$
  Train $\mathcal{M}_\theta$ on $\mathcal{L}$ to get a new $\mathcal{M}_{\theta'}$
until convergence or max iteration is reached

4 Experiments

As described in the introduction, we aim to propose a method for improving the reasoning of LLMs beyond the languages. Hence, we operate on multilingual reasoning tasks. We use three models (§4.1) trained as described in §4.2 and evaluated two mathematical reasoning tasks (§4.3) using the configurations described in §4.4.

4.1 Models

To conduct our study on different models and have a term of comparison, we use Llama3-8B (Grattafiori et al., 2024), Phi-3.5-mini (Abdin et al., 2024) and DeepSeekMath-7B-Instruct (Shao et al., 2024b) (DeepSeek-7B). Furthermore, to show the scalability and effectiveness of our approach on further models, we introduce additional evaluations. Details are reported in Appendix J.

4.2 Training Methods

As introduced in §3, we use a Self-tuning technique based on iterative steps of SFT and RL. We follow standard practice and perform a warm-up phase based on an SFT step using synthetic demonstrations discussed in §4.3.2. Then, we conduct the Self-training by progressively applying SFT and RL optimisation algorithms (DPO and GRPO, as in Algorithm 1) in an iterative manner. Following pilot studies (later discussed), we set the total number of iterations to three (excluding warm-up), the same for the settings where we use only one between SFT and RL.

Preference Optimisation RL We employ the HuggingFace trainers ($DPO_{trainer}$ and $GRPO_{trainer}$) to ensure reproducibility. For DPO, we set the learning rate to *1e-6* and β to 0.1. The optimisation process is set at a maximum of 2000 steps, saving the checkpoint corresponding to the lowest validation loss. For GRPO, we set the learning rate to *5e-6* and β to x. The optimisation process is set at a maximum of 2000 steps, saving the checkpoint corresponding to the lowest validation loss. Details in Appendix I.

Supervised Fine-tuning Regarding the SFT phase, we employed 8-bit quantization and LoRA. We tune the model for one epoch (warm-up) and for one epoch for each iteration using the learning rates according to the specific model configuration, as detailed in Appendix I.

4.3 Data

4.3.1 Evaluation Set

To study the multilingual reasoning performances of trained models, we operate via MGSM, MSVAMP, and we introduce MGSM-SYMBOLIC.

Mathematical Reasoning task We use the extension of GSM8K and SVAMP. Respectively, Multilingual Grade School Math (MGSM) and Multilingual Simple Variations on Arithmetic Math word Problems (MSVAMP). In original cases, the authors proposed a benchmark of English mathematical problems with the following structure: a word problem in natural language and a target answer in numbers. For both versions, a subset of instances from the official list of examples were translated into 11 different languages, maintaining the structure of the input and output.

MGSM-SYMBOLIC Mirzadeh et al. (2024) improved GSM8k (the ancestor of MGSM) by proposing GSM-Symbolic. This introduces symbolic patterns in GSM8k that complexify the task and disadvantage the LLMs' capabilities. We propose mGSM-Symbolic, the multilingual GSM-Symbolic extension. In particular, we conduct an automatic translation phase disillusioned by qualified annotators in 10 different languages. The dataset is available on **GitHub**[1] and **HuggingFace**[2].

Evaluation Metrics To evaluate the performances, we use the accuracy of the final answer, assessed through an exact match between the generated response and the ground truth. Additionally, to observe the dynamics emerging across different languages, we use the Answer Consistency Ratio (ACR), which is defined as the ratio between the set of math questions answered correctly in English m and those answered correctly beyond English m, calculated as: $ACR = \frac{|m \cap n|}{|n|}$. A higher ACR indicates a greater degree of overlap in reasoning capabilities among the languages.

4.3.2 Training Set

Instead of using natural language rationale, we employ synthetic demonstrations to train models to solve tasks following the two phases in Figure 1. Specifically, we instruct a robust model capable of addressing multilingual mathematical tasks by formalising problems and solving them in a language-

[1] lranaldii/MGSM-Symbolic

[2] lrana/MGSM-Symbolic

agnostic manner. We employ GPT-4o as annotator, instructing it with the prompt detailed in Appendix A (we define this procedure as `Self-training`)

Different works train an expert version of the same model that is going to be refined for generating synthetic demonstrations, which are subsequently used for self-training (we define this procedure as `Full Self-training`).

Multilingual Demonstrations We annotate a subset of the MSVAMP dataset containing 250 samples for all languages to have in-domain demonstrations. After the annotation process, we check the quality of the demonstrations using rule-based heuristics and GPT-4o-mini as an additional evaluator (details in Appendix F).

4.4 Experimental Setup

We test the following configurations:

In-context Learning We evaluate the baseline models (without tuning) using a few-shot strategy (6-shot) defined as `Direct` and `CoT`. Moreover, we instruct the models to solve the problem following SWATH.

Training We assess the impact of the Self-training approaches (§4) by conducting different tuning configurations:

- **SFT, RL** We tune the models using the synthetic demonstrations as detailed in Appendix E.
- **Self-training** We warm-up the models using the synthetic demonstrations as detailed and conduct the self-training strategies using both policies.
- **FULL Self-training** Finally, to observe the impact of the self-generated demonstrations, we conduct both the annotation, SFT (warm-up) and FULL Self-train phase completely on the self-generated data of the same expert model.

5 Results

Reasoning can be performed via language-agnostic formalisms that large language models (LLMs) can employ to enhance performance in multilingual tasks. SWATH that leads to a formal symbolic reasoned resolution, directing LLMs to deliver robust answers across different languages. While SWATH is effective in GPT-4o, in smaller models, it does not have the same benefits. Self-training allows smaller models to deliver formal reasoning trajectories while gaining the same benefits without operating through instructions and achieving more consistent results than GPT-4o (§5.1). Self-training is definitely more performant than single SFT or RL and allows the models to achieve better results with significantly fewer training data (§5.2). In in-depth studies, we analyse the emerging dynamics between languages (§5.3) and demonstrate the scalability of our method operating the Self-training in further models (Appendix K).

5.1 Reasoning in Abstractive Spaces

The operation of our formalisation method positively influences the models' performance in multilingual reasoning obtaining substantial benefits on the proposed tasks.

Models	AVG	EN	SW
GPT-4o	69.2 (-1.7)	83.2 (-3.6)	70.5 (-3.4)
+SWATH	84.2 (-0.2)	93.0 (-0.8)	84.4 (-0.6)
Llama3-8B	54.8 (-3.4)	76.0 (3.6)	55.2 (-6.6)
+`Self-training`	73.0(**-1.0**)	91.8 (-0.6)	71.2 (-0.6)
Phi-3.5	49.9 (-2.6)	65.6 (-2.4)	52.6 (-2.2)
+`Self-training`	63.0(**-1.1**)	81.6(**-1.0**)	63.2 (-0.4)
DeepSeek-7B	55.2(**-1.6**)	76.2 (-2.2)	54.0 (-2.2)
+`Self-training`	73.0 (-0.8)	90.2 (-0.4)	71.4 (-0.6)

Table 1: Performances on MGSM-SYMBOLIC in brackets the differences between MGSM.

Multilingual Symbolic Reasoning Table 1 shows that SWATH with GPT-4o achieves consistent results in MGSM-SYMBOLIC, in line with those obtained in MGSM (values in brackets). In addition, the Self-training approach benefits the abstraction capabilities of the tuned models as they are not affected by the biases introduced in MGSM-SYMBOLIC. In contrast, the baseline strategies obtains definitely lower results. This demonstrates the functionality of SWATH's formalisation process.

In-context Learning Table 2 reports the results of SWATH on GPT-4o, compared to previous prompting-based methods (`Direct`, `CoT`), it performs better (+18.2% compared to `Direct`, +13.9% compared to `CoT`). The nature of in-context instructions leads the models to formalise problem solutions systematically and encourages planning by improving the evolution of reasoning. As a result, the reasoning trajectories are more consistent and not language-specific, avoiding performances of imbalances. On the other hand, in smaller models such as Llama-3-8B, Phi-3.5-mini, and DeepSeek-7B, we observe a decrease in performance (see red values in brackets reported in Table 2), indicating

Model	Method	MGSM		MSVAMP		Average	
		ACC	ARC	ACC	ARC	ACC	ARC
GPT-4o	Direct	70.9	46.8	69.2	44.2	70.1	40.5
	CoT	73.5	55.6	72.2	56.9	72.8	56.2
	SWATH	**84.2**	**68.9**	**82.7**	**69.3**	**83.4**	**69.1**
Llama-3-8B	Direct(SWATH as ICL)	58.2 (57.3)	30.8	62.6 (61.0)	34.3	60.4 (59.1)	32.5
	RL (GRPO)	66.0	53.7	67.8	55.2	66.9	54.4
	SFT	62.3	52.4	65.0	50.1	63.6	49.7
	Self-training	**74.0**	**69.2**	**71.5**	**70.6**	**72.7**	**69.9**
Phi-3.5-mini	Direct(SWATH as ICL)	52.5 (53.7)	32.8	57.9 (56.4)	36.4	55.2 (55.0)	34.6
	RL (GRPO)	58.5	36.2	62.2	44.8	60.3	40.5
	SFT	58.1	34.9	59.4	38.6	58.7	36.6
	Self-training	**64.1**	**48.3**	66.0	**49.5**	**65.0**	**48.9**
DeepSeek-7B	Direct(SWATH as ICL)	56.8 (56.2)	38.6	63.0 (62.3)	39.2	60.0 (59.4)	38.9
	RL (GRPO)	68.2	59.0	70.8	63.8	69.5	61.4
	SFT	62.9	57.7	67.2	60.2	65.0	58.9
	Self-training	**73.8**	**72.5**	**74.8**	**70.2**	**74.9**	**72.3**

Table 2: Average accuracy (**ACC**) and Answer Consistency Ratio (**ARC**) scores using methods introduced in §3. We report the models trained via the GRPO algorithm (in Table 7, we compare DPO and GRPO). *(in **bold** the best performance per model, in brackets the SWATH as in-context learning strategy detailed in Table 7).

that such models are unable to follow the instructions and do not fully benefit from this approach.

Self-training Table 2 and detailed in Appendix D report the results of Self-training for different models. From the results, it clearly emerges that Self-training is consistently effective in enhancing the performance of the proposed models. Although Self-trained models do not outperform GPT-4o, they achieve greater consistency between languages (answer consistency ratio, i.e., ARC). In particular, concerning ARC, we observe a robust gain over the GPT-4o (with an improvement of 2% for Llama-3-8B and 4.6% for DeepSeek-7B).

		MGSM	MSVAMP	Avg
Llama-3-8B	**RL**	+3.6	+2.8	+3.2
	SFT+RL	**+11.2**	+3.9	**+7.5**
Phi-3.5	**RL**	+0.5	+1.6	+1.0
	SFT+RL	+1.4	+3.4	+2.4
DeepSeek-7B	**RL**	+5.8	+4.4	+5.1
	SFT+RL	**+12.1**	+6.0	**+9.0**

Table 3: Differences (Δ) between GRPO and DPO when used alone (denoted as RL) and in Self-training settings (denoted as SFT+RL). *In **bold**, the highest differences.

5.2 The Self-training Impact

The Self-training process delivers more robust models by consistently increasing performance and using less training data than the other tuning approaches, including SFT and RL. In Table 2, we observe general robust improvement over the SFT with an improvement of 9.6% for Phi-3.5, 12.6% for Llama-3-8B and 13.5% for DeepSeek-7B. The improvements are likewise in terms of data consumption efficiency, as Self-tuning operates using a reduced number of displays compared to SFT configurations, as reported in Appendix G.

The role of RL In Table 2, we reported the results achieved using GRPO. Table 3 shows that GRPO consistently outperforms DPO when used independently and in synergy with SFT (complete Self-training). As described in §3.1, the GRPO metric is not opted via an annotated dataset. As in previous contributions, a rule-based algorithm can be defined as a reward model. Hence, GRPO fits the proposed task instead of operating on single instances, and it aims to optimise groups of instances that, by construction, are in different languages.

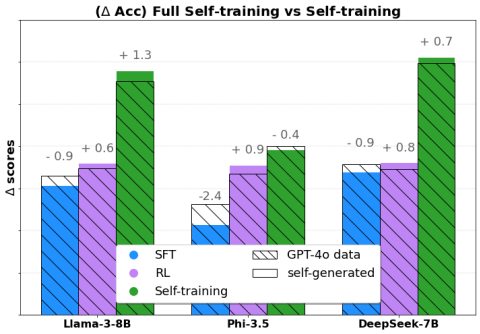

Figure 2: Average differences using data generated by GPT-4o and self-generated (i.e. FULL Self-training).

The impact of FULL Self-training Current alignment policies operate via demonstrations gen-

erated by an expert model from the same families. Ranaldi and Freitas (2024a) reveal that- in-family learning has a more significant impact on student models. We applying FULL Self-tuning, and we show that self-generated demonstrations could benefit models more robustly than data generated by GPT-4o. Figure 2 shows that self-generated annotations result in more robust and consistent models across languages for the same amount of training.

5.3 Low-resource Language Improvements

Reasoning is language-agnostic; however, natural language explanations are used to structure and externalise reasoning processes. The SWATH formalism supports multilingual reasoning consistently across languages and achieves the same benefits in high-resource (HR) and low-resource (LR) languages. Figure 3 shows the improvement over baselines in GPT-4o and the improvement of Self-trained models over baselines and models tuned only via SFT or RL. SWATH as in-context approach used with GPT-4o achieves higher accuracy in HR than in LR (because of the bias of the languages themselves). However, comparing SWATH's performances with baseline we observe a major improvement in low-resource languages (+12.7 in LR and +9.2 in HR). We observe consistent benefits regarding the Self-trained models as well. Although they do not outperform GPT-4o, they achieve the same improvement when compared to baseline models. Moreover, in the LR, they even achieve significant benefits that exceed +17.5 points (except Phi-3.5, which achieves +6.8).

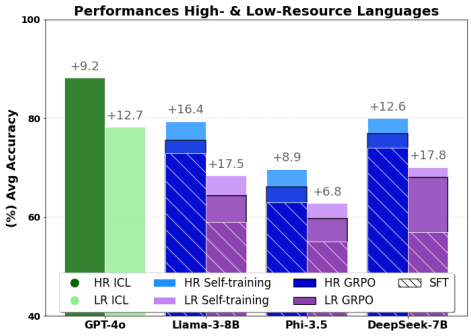

Figure 3: SWATH performances in High- and Low-resource Languages. In GPT-4o, it is used as ICL in the other models via Self-training, SFT and RL (GRPO). *(The differences with the baselines above the bars).

This demonstrates that SWATH: *(i)* disentangling logical reasoning and content obtains substantial benefits both in the more robust models capable of performing this abstraction step and *(ii)* in the smaller models, not able to perform abstraction via direct instruction provided at inference-phase, consistently improves results by providing performance quasi-alignment between languages.

6 Conclusion

Although reasoning is language-agnostic, the LLMs' generations are unbalanced towards dominant languages in pre-training. In fact, while LLMs have a good level of proficiency in different languages and tasks, the ability to deliver step-wise reasoned responses remains unstable. To make multilingual reasoning more equitable and robust across languages, we propose a modular approach that first instructs the model to formalise the problem in an abstractive problem space and then solve it using step-wise passages. In order to align performance across languages, we operate through a self-training approach and show that this substantially improves performance by making multilingual reasoning trajectories more correct.

Limitations & Future Work

In future developments, we plan to extend the analysis to other tasks and take care of the efficiency of the methodologies used. In particular, with regard to tasks, we would like to expand the analysis to tasks of common-sense reasoning, natural language comprehension and inference in multilingual spaces. Regarding improving efficiency, we would like to investigate the use of FULL operating with reduced computational and data resources.

In our study, we assessed the efficacy of our method using GPT-based models (closed-source) and different open-source models. Moving forward, it will be crucial to explore how our methods perform comparative to other closed-source large language models. Given the constraints of evaluation benchmarks and the expenses associated with the OpenAI API, our testing was restricted to a limited set of tasks and languages, merely touching upon the global linguistic diversity. Furthermore, while our study has considered and evaluated different models, we aim to more thoroughly examine the outcomes of models pre-trained specifically for individual languages (language-focused). Currently, there are few open resources of comparable scale to those we have examined. We hope that such models become more accessible in the future, allowing for a deeper investigation into this area.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Qin Cai, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Yen-Chun Chen, Yi-Ling Chen, Parul Chopra, Xiyang Dai, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Victor Fragoso, Dan Iter, Mei Gao, Min Gao, Jianfeng Gao, Amit Garg, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Ce Liu, Mengchen Liu, Weishung Liu, Eric Lin, Zeqi Lin, Chong Luo, Piyush Madan, Matt Mazzola, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Xin Wang, Lijuan Wang, Chunyu Wang, Yu Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Haiping Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Sonali Yadav, Fan Yang, Jianwei Yang, Ziyi Yang, Yifan Yang, Donghan Yu, Lu Yuan, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone.

Marie Amalric and Stanislas Dehaene. 2016. Origins of the brain networks for advanced mathematics in expert mathematicians. *Proceedings of the National Academy of Sciences*, 113(18):4909–4917.

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

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.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.

Vedant Gaur and Nikunj Saunshi. 2023. Reasoning in large language models through symbolic math word problems. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5889–5903, Toronto, Canada. Association for Computational Linguistics.

Gayane Ghazaryan, Erik Arakelyan, Isabelle Augenstein, and Pasquale Minervini. 2025. SynDARin: Synthesising datasets for automated reasoning in low-resource languages. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 6459–6466, Abu Dhabi, UAE. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, and inter alia. 2024. The llama 3 herd of models.

Haoyang Huang, Tianyi Tang, Dongdong Zhang, Wayne Xin Zhao, Ting Song, Yan Xia, and Furu Wei. 2023. Not all languages are created equal in llms: Improving multilingual capability by cross-lingual-thought prompting.

Chong Li, Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2024. Improving in-context learning of multilingual generative language models with cross-lingual alignment. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8058–8076, Mexico City, Mexico. Association for Computational Linguistics.

Yong Lin, Skyler Seto, Maartje Ter Hoeve, Katherine Metcalf, Barry-John Theobald, Xuan Wang, Yizhe Zhang, Chen Huang, and Tong Zhang. 2024. On the limited generalization capability of the implicit reward model induced by direct preference optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16015–16026, Miami, Florida, USA. Association for Computational Linguistics.

Kyle Mahowald, Anna A. Ivanova, Idan A. Blank, Nancy Kanwisher, Joshua B. Tenenbaum, and Evelina Fedorenko. 2024. Dissociating language and thought in large language models.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models.

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.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang. 2023. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3806–3824, Singapore. Association for Computational Linguistics.

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.

Leonardo Ranaldi and Andre Freitas. 2024a. Aligning large and small language models via chain-of-thought reasoning. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1812–1827, St. Julian's, Malta. Association for Computational Linguistics.

Leonardo Ranaldi and Andre Freitas. 2024b. Self-refine instruction-tuning for aligning reasoning in language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2325–2347, Miami, Florida, USA. Association for Computational Linguistics.

Leonardo Ranaldi and Giulia Pucci. 2023. Does the English matter? elicit cross-lingual abilities of large language models. In *Proceedings of the 3rd Workshop on Multi-lingual Representation Learning (MRL)*, pages 173–183, Singapore. Association for Computational Linguistics.

Leonardo Ranaldi, Giulia Pucci, and Andre Freitas. 2024a. Empowering cross-lingual abilities of instruction-tuned large language models by translation-following demonstrations. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7961–7973, Bangkok, Thailand. Association for Computational Linguistics.

Leonardo Ranaldi, Giulia Pucci, Barry Haddow, and Alexandra Birch. 2024b. Empowering multi-step reasoning across languages via program-aided language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12171–12187, Miami, Florida, USA. Association for Computational Linguistics.

Leonardo Ranaldi, Giulia Pucci, Federico Ranaldi, Elena Sofia Ruzzetti, and Fabio Massimo Zanzotto. 2024c. A tree-of-thoughts to broaden multi-step reasoning across languages. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1229–1241, Mexico City, Mexico. Association for Computational Linguistics.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024a. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024b. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

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.

Ahmet Üstün, Viraat Aryabumi, Zheng Yong, Wei-Yin Ko, Daniel D'souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, Freddie Vargus, Phil Blunsom, Shayne Longpre, Niklas Muennighoff, Marzieh Fadaee, Julia Kreutzer, and

Sara Hooker. 2024. Aya model: An instruction finetuned open-access multilingual language model. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15894–15939, Bangkok, Thailand. Association for Computational Linguistics.

Tianduo Wang, Shichen Li, and Wei Lu. 2024. Self-training with direct preference optimization improves chain-of-thought reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11917–11928, Bangkok, Thailand. Association for Computational Linguistics.

Dongkeun Yoon, Joel Jang, Sungdong Kim, Seungone Kim, Sheikh Shafayat, and Minjoon Seo. 2024. LangBridge: Multilingual reasoning without multilingual supervision. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7502–7522, Bangkok, Thailand. Association for Computational Linguistics.

Yuanchi Zhang, Yile Wang, Zijun Liu, Shuo Wang, Xiaolong Wang, Peng Li, Maosong Sun, and Yang Liu. 2024. Enhancing multilingual capabilities of large language models through self-distillation from resource-rich languages. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11189–11204, Bangkok, Thailand. Association for Computational Linguistics.

A SWATH Instruction Template

#Role
You are an experienced expert skilled in multilingual mathematical reasoning problems.

#Task
You are presented with a mathematical reasoning problem in a given language. Follow the steps below rigorously to formalize and solve it.

#Instructions

1) Formalisation (Language-Agnostic): Identify and define the key mathematical components of the problem, such as variables, functions, operations, and constraints. Structure these components in an abstract manner to ensure a clear and precise formulation. *Label this step as <formalisation>....</formalisation>*

2) Reasoning Execution: Solve the problem systematically by breaking it into logical steps. Clearly justify each step using natural language explanations while maintaining logical rigor. Express the final answer in the same language as the input query. *Label this step as <reasoning>....</reasoning>*

Final Answer: Present the extracted answer in a concise format, marked as "**The answer is: [num]**" in the same language as the query. *Label this step as <answer>....</answer>*

#Question
{`question`}

Table 4: The SWATH instructs the model to abstract problem components and deliver step-wise reasoning paths that lead the models to solve multilingual task.

B Results Arithmetic Reasoning Tasks GPT-4o

Model	Method	en	de	zh	fr	ru	sw	es	bn	ja	te	th	Avg
							MGSM						
GPT-4o	Direct	86.8	78.0	79.2	83.0	78.4	76.2	82.2	38.8	72.0	40.4	65.4	70.9
	CoT	92.5	78.8	79.6	84.2	79.2	77.2	83.4	44.0	76.2	45.4	66.2	73.5
	SWATH	93.0	90.2	89.4	88.6	87.5	85.0	88.8	79.0	84.4	70.6	74.6	84.2
							MSVAMP						
GPT-4o	Direct	83.2	74.1	73.6	81.2	76.3	70.5	77.2	36.0	70.5	-	65.9	69.2
	CoT	89.8	74.6	74.2	81.8	76.2	71.4	78.1	38.0	71.2	-	66.3	72.2
	SWATH	96.2	85.7	82.1	87.1	81.9	80.4	86.4	72.7	78.2	-	76.6	82.7

Table 5: Accuracies (%) on MGSM and SVAMP of GPT-4o on first 100 questions for each language..

C Results on MGSM-SYMBOLIC

Model	Method	en	de	zh	fr	ru	sw	es	bn	ja	te	th	Avg
GPT-4o	Direct	83.2	74.4	75.4	79.8	74.6	72.8	78.8	35.2	68.6	37.2	61.6	67.3
	CoT	89.4	75.8	76.6	81.2	76.2	73.8	80.4	41.0	73.0	42.2	62.8	70.2
	SWATH	92.2	89.4	88.6	88.0	87.2	84.2	88.0	78.4	84.0	69.2	74.4	84.0
Llama-3-8B	Direct	76.0	57.8	58.0	58.0	56.4	55.2	60.8	47.0	48.8	43.2	42.0	54.8
	Self-training	91.8	83.0	74.0	73.2	72.4	71.2	76.4	64.8	64.0	65.8	65.2	73.0
Phi-3	Direct	65.6	55.8	56.0	54.8	54.8	52.6	55.2	45.0	46.6	28.4	34.8	49.9
	Self-training	81.6	64.8	62.8	65.8	64.8	63.8	64.4	55.2	60.0	56.0	54.8	63.0
DeepSeek-7B	Direct	75.8	57.6	67.8	57.8	56.0	54.0	60.8	45.4	49.0	43.3	38.8	55.2
	Self-training	90.2	82.0	80.4	72.4	72.2	71.4	75.0	65.3	63.0	65.8	65.0	73.0

Table 6: Accuracies (%) on MGSM-SYMBOLIC using baseline (`Direct`) and `Self-training` via GRPO.

D Results Arithmetic Reasoning Tasks

Model	Method	en	de	zh	fr	ru	sw	es	bn	ja	te	th	Avg
							MGSM						
Llama-3-8B	Direct	79.6	61.0	61.4	61.2	59.8	58.8	64.4	50.4	52.0	46.6	45.4	58.2
	SWATH (ICL)	78.6	60.5	60.5	60.1	58.7	57.7	63.9	49.2	51.0	45.7	44.6	57.3
	SFT	82.6	64.4	64.6	64.4	62.8	62.2	67.8	53.4	55.0	49.8	48.8	61.5
	Full SFT	81.6	63.0	63.5	63.2	61.2	61.0	66.8	52.4	54.2	48.8	47.6	60.4
	RL$_{(DPO)}$	83.6	66.0	63.0	65.2	61.0	62.8	68.6	52.0	56.0	50.5	44.6	62.2
	RL$_{(GRPO)}$	84.0	76.2	67.2	66.4	65.6	63.0	69.6	58.4	57.2	59.0	58.0	66.0
	Self-training$_{(DPO)}$	86.4	69.2	66.0	68.0	64.5	65.7	71.4	55.8	58.8	53.3	47.4	62.4
	Full Self-training$_{(DPO)}$	84.6	67.0	64.0	65.8	62.8	63.6	69.2	54.0	57.0	51.4	45.8	62.0
	Self-training$_{(GRPO)}$	92.4	84.0	75.2	74.2	73.4	71.8	77.6	66.0	65.0	67.0	66.2	74.0
	Full Self-training$_{(GRPO)}$	94.0	85.8	77.0	75.8	75.4	73.2	79.4	67.8	67.2	69.2	68.6	75.8
Phi-3-mini	Direct	68.0	58.2	58.5	57.4	57.1	55.0	57.8	47.5	49.2	30.9	37.6	52.5
	SWATH (ICL)	75.1	57.8	56.8	58.8	57.0	56.3	57.1	47.7	49.4	37.7	36.9	53.7
	SFT	79.6	62.6	61.0	63.1	61.0	60.5	61.4	51.6	53.5	42.8	41.0	58.1
	Full SFT	76.2	59.6	57.0	60.0	58.2	57.4	58.0	48.2	50.4	39.8	38.0	55.7
	RL$_{(DPO)}$	79.0	61.2	61.0	62.0	61.2	60.2	61.4	52.4	52.2	45.0	44.2	58.0
	RL$_{(GRPO)}$	75.6	60.9	59.0	61.6	60.8	59.0	60.8	51.4	56.0	46.9	45.6	58.5
	Self-training$_{(DPO)}$	81.2	64.0	63.2	65.0	63.4	62.4	63.6	54.6	55.0	47.0	46.2	60.7
	Full Self-training$_{(DPO)}$	79.1	62.0	61.0	63.0	61.6	60.6	61.6	52.8	53.0	45.5	44.3	58.6
	Self-training$_{(GRPO)}$	82.6	65.8	64.0	66.8	65.8	64.2	65.4	56.4	61.2	57.0	55.8	64.1
	Full Self-training$_{(GRPO)}$	81.3	64.0	63.4	65.2	63.5	62.5	63.4	54.8	59.4	55.0	54.2	62.1
DeepSeek-7B	Direct	78.0	59.7	69.7	59.9	58.2	56.2	62.8	47.6	50.9	45.4	40.7	56.8
	SWATH (ICL)	77.1	59.1	69.0	59.0	57.5	56.3	62.0	47.0	50.2	44.8	39.8	56.2
	SFT	84.1	66.0	65.6	68.2	64.6	63.4	69.3	54.8	56.8	51.4	50.2	62.9
	Full SFT	82.1	66.7	69.9	64.7	63.3	62.0	68.2	53.6	55.8	50.0	50.2	62.0
	RL$_{(DPO)}$	84.0	67.2	67.0	65.2	62.0	63.4	69.4	53.6	57.2	51.2	45.0	62.4
	RL$_{(GRPO)}$	86.9	78.0	77.0	69.2	68.8	68.9	72.2	62.3	58.2	62.6	62.2	68.2
	Self-training$_{(DPO)}$	83.0	65.8	68.2	65.0	61.4	63.0	68.6	53.4	55.6	49.8	44.6	61.7
	Full Self-training$_{(DPO)}$	86.1	68.0	70.7	67.5	63.6	65.2	70.8	55.8	57.8	52.2	47.0	63.8
	Self-training$_{(GRPO)}$	90.8	82.6	81.6	73.2	72.8	72.0	76.0	66.2	63.8	66.5	66.0	73.8
	Full Self-training$_{(GRPO)}$	92.8	84.6	83.0	75.0	74.5	71.8	79.0	66.9	66.2	68.4	67.0	74.5
							MSVAMP						
Llama-3-8B	Direct	81.3	62.8	62.9	62.8	61.4	60.4	66.1	52.1	53.8	-	48.3	62.6
	SWATH (ICL)	81.5	63.3	62.7	61.9	61.3	60.5	66.1	52.0	53.7	-	47.2	61.0
	SFT	85.0	66.4	66.6	66.7	65.0	63.8	69.5	55.2	57.4	-	51.5	65.0
	Full SFT	85.6	67.3	68.1	67.7	66.1	65.1	70.6	56.6	58.7	-	53.0	65.9
	RL$_{DPO}$	82.5	66.3	66.8	66.6	65.3	63.6	69.8	53.2	60.3	-	55.7	65.0
	RL$_{GRPO}$	85.1	68.6	68.8	69.1	67.5	65.8	71.6	60.0	63.0	-	60.9	67.8
	Self-training$_{DPO}$	85.2	68.8	69.0	69.2	67.6	66.2	72.0	55.7	62.9	-	58.0	67.6
	Full Self-training$_{DPO}$	82.5	66.1	66.3	66.5	64.9	63.5	69.3	53.0	60.2	-	55.3	65.8
	Self-training$_{GRPO}$	88.4	72.0	72.3	72.5	71.0	69.6	75.4	63.2	66.2	-	64.4	71.5
	Full Self-training$_{GRPO}$	88.6	71.4	75.4	73.1	70.2	69.3	75.1	63.7	65.6	-	64.6	71.7
Phi-3-mini	Direct	77.9	59.4	59.7	59.6	58.2	57.2	62.9	48.9	50.7	-	44.8	57.9
	SWATH (ICL)	77.0	58.9	56.6	59.7	58.5	56.9	62.5	48.4	50.3	-	44.5	56.4
	SFT	78.9	61.2	61.3	61.4	60.1	58.6	64.3	50.0	52.2	-	46.4	59.4
	Full SFT	81.3	63.6	63.6	63.8	62.5	61.0	66.7	52.4	54.5	-	48.7	59.1
	RL$_{DPO}$	77.6	62.3	62.4	62.7	61.1	59.6	65.6	49.2	56.4	-	51.4	60.8
	RL$_{GRPO}$	78.1	62.7	63.0	63.2	61.6	60.2	66.2	53.7	56.8	-	55.2	62.2
	Self-training$_{DPO}$	81.5	66.2	66.3	66.6	65.0	63.5	69.5	53.1	60.3	-	55.3	64.7
	Full Self-training$_{DPO}$	79.2	63.9	63.8	64.0	62.6	60.7	67.1	50.7	57.6	-	53.0	62.3
	Self-training$_{GRPO}$	82.0	66.6	66.9	67.1	65.5	64.1	70.1	57.6	60.7	-	59.1	66.0
	Full Self-training$_{GRPO}$	79.2	64.6	65.1	65.0	63.4	61.2	68.1	55.7	58.6	-	56.5	63.9
DeepSeek-7B	Direct	83.0	64.5	74.6	64.5	63.0	62.1	67.8	53.8	55.5	-	50.0	63.0
	SWATH (ICL)	82.3	64.2	68.9	64.1	62.6	61.5	67.1	52.9	54.9	-	49.3	62.3
	SFT	86.6	68.1	70.2	68.4	66.7	65.5	71.1	56.8	59.1	-	53.2	67.2
	Full SFT	88.0	69.4	73.3	69.9	68.2	67.3	72.8	58.7	61.1	-	55.3	68.1
	RL$_{DPO}$	83.4	67.1	69.2	67.7	66.0	64.7	70.3	53.9	61.3	-	56.3	66.4
	RL$_{GRPO}$	87.1	70.8	75.7	71.1	69.1	68.3	74.1	61.8	67.7	-	63.1	70.8
	Self-training$_{DPO}$	87.3	71.0	73.1	71.6	69.9	68.6	74.2	57.8	65.2	-	60.2	69.0
	Full Self-training$_{DPO}$	84.8	68.4	71.5	68.7	67.2	65.7	71.6	55.1	62.4	-	57.7	67.0
	Self-training$_{GRPO}$	90.9	74.5	79.8	75.0	73.2	72.1	78.0	65.6	71.6	-	67.0	74.8
	Full Self-training$_{GRPO}$	90.6	71.5	78.4	73.0	76.5	70.2	74.1	60.9	68.6	-	62.9	75.5

Table 7: Accuracies (%) on MGSM and SVAMP of using the methods described in §3.

E Annotations Pipeline

We use SWATH to generate synthetic demonstrations for training smaller LLMs. We use GPT-4o as an annotator and use the annotations to warm-up the models with the proposed methodologies. We then conduct a complete Self-training phase. Moreover, we conduct the Self-training by using self-generated data (generated by the trained models themselves). We define these configurations 'FULL'-Self-training. In both cases, the demonstrations are generated by prompting the models using instructions detailed in Appendix A. However, while GPT-4o follows the instructions well (in fact, we did not find any significant issues), the other models generate outcomes that include errors. To handle this, we evaluated the quality of the generated demonstrations by filtering out inaccurate examples to get a gold instruction set. In particular, we removed all inaccurate answers (outputs that do not match the exact target string metric). Then, we control if the demonstrations follow correctly the steps indicated in our prompt (see Table 4) using GPT-4o-mini and the prompt in Appendix F.

F Evaluation Metrics

We used a double-check to assess the accuracy of the responses delivered in the different experiments. In the first step, we used an exact-match heuristic. However, since some experiments required a more accurate response check, we used GPT-4o-mini as a judge. Hence, we prompt the model as follows:

GPT-4o Evaluation Prompt

#Role:
You are an experienced expert skilled in answering complex problems through logical reasoning and structured analysis.
#Instructions:
Given the following "#Senteces", you are a decider that decides whether the "Generated Answer" follows the "Required Format" and the final answer is the same as the "Target Answer". If the output doesn't align with the required format and target answer, respond with '0', whereas if it's correct, then respond with '1'. *Please, do not provide any other answer beyond '0' or '1'.*
#Senteces:
Generated Answer: {model_result}
Required Format: {format}
Target Answer: {correct_answer}.

G Data Composition

As evaluation sets, we use the tasks introduced in §4.3. These tasks are used to assess the performance of LLMs, but they do not have reserved sets for evaluation and training. Therefore, to produce a training set, we split MSVAMP into training and testing. Table 8 shows the instances of each dataset in training and testing. To ensure the languages are perfectly balanced, we translated 350 samples from English to Telugu (language non-present in MSVAMP). This subset was used for training purposes only.

Task	Total	Test	Train. Set	# dim
MGSM	$2.5k$	$2.5k$	No	No
MGSM-SYMBOLIC	$2.5k$	$2.5k$	No	No
MSVAMP	$10k$	$2.5k$	Yes	$3,5k$

Table 8: Training and evaluation data. *($1k$ is equal to 1000).

The data are perfectly balanced between the languages in the proposed tasks. However, as described in Appendix E, the qualities of the annotations are not perfect. Behind filtering the annotations, we obtained a reduced dataset, respectively 1.6k for GPT-4o and about 1.2k for the other models. To have fair, balanced subsets, we use 1k samples in total. We use 1k samples when instructing the models for DPO and SFT. For the Self-training, we used as the initial subset (§3.2) 60% of the filtered samples balanced between all languages.

H Number of Iterations

Following pilot experiments, we set the number of iterations of self-tuning at three. Figure 9 shows the performances trend by increasing number of iterations, epochs and steps after warm-up (wup).

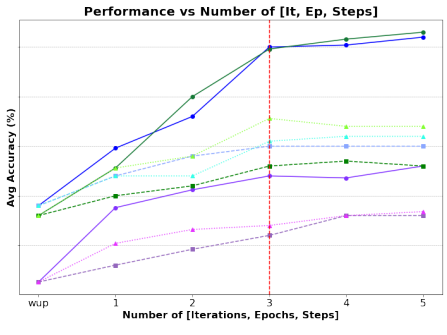

Table 9: Average accuracies on MGSM.

I Models and Hyperparameters

Models In our experimental setting, as introduced in §4.1, we propose different models (detailed in Table 10). Finally, we fix generation temperature to $\tau = 0$ for all models. We choose these temperature for (mostly) deterministic outputs, with a maximum token length of 512. The other parameters are left unchanged as recommended by the official resources. We use four 48GB NVIDIA RTXA600 GPUs for all experiments performed only in inference.

Hyperparameters In §4.2, we described the standard Self-training setting. However, we have proposed different experimental settings. In the Self-training experimental setting, we conducted three iterations as proposed in (Wang et al., 2024; Shao et al., 2024b). In the SFT-only and RL-only settings, we used warm-up and four epochs and 8000 steps, respectively. We conducted this setting after the pilot experiments shown in the previous sections.

J Models Vesions

Model	Version
Llama3-8(-instruct)	meta-llama/Meta-Llama-3-8B-Instruct
Phi-3(-mini-instruct)	microsoft/Phi-3-mini-4k-instruct
DeepSeekMath-7B	deepseek-ai/deepseek-math-7b-instruct
GPT-4o	gpt-4o-2024-08-06
GPT-4o-mini	gpt-4o-mini-2024-07-18

Table 10: List the versions of the models proposed in this work, which can be found on huggingface.co. We used all the default configurations proposed in the repositories for each model.

K Transferability in Smaller Models

To analyse the transferability of Self-training and SWATH, we extend the experiments to smaller models exemplified by `Llama-3-1B`, `EuroLLM-1.7B` and `Velvet-2B` chosen for the purpose of construction, i.e. their multilingual nature, for the performances obtained in mathematical reasoning tasks and for the reduced number of parameters that allowed different experiments to be conducted. Hence, we adopt the experimental setup proposed in §4.1 using SFT, GRPO and Self-training. Table 11 shows the average scores obtained on MGSM-SYMBOLIC. On average Self-training via SWATH, outperforms the baselines, RL and SFT models resulting in an ARC comparable to that obtained by Llama-3-8B and DeepSeek-7B in Table 2.

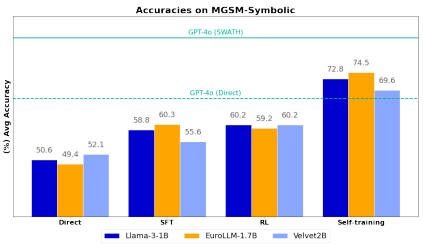

Table 11: Average accuracies of smaller models in our MGSM-SYMBOLIC.

L Proposed Task

Dataset	Languages	#Lang
MGSM	Bengali (bn), Chinese (zh), French (fr), Thai (th), German (de), Japanese (jp), Russian (ru), Telugu (te), Spanish (es), Swahili (sw)	11
MSVAMP	Bengali (bn), Chinese (zh), French (fr), Thai (th), German (de), Japanese (jp), Russian (ru), Spanish (es), Swahili (sw)	11
MGSM-SYMBOLIC	Bengali (bn), Chinese (zh), French (fr), Thai (th), German (de), Japanese (jp), Russian (ru), Telugu (te), Spanish (es), Swahili (sw)	12

Table 12: Languages present in datasets used in this work.

M GRPO Tuning

This approach evaluates model-generated completions using **five key constraints**:

1. Strict Reward (r_s) Ensures factual accuracy by comparing the extracted answer to the ground truth. The reward is computed as:

$$r_s(y) = \begin{cases} 2 & \text{if } \texttt{extract\_match}(y) = \hat{y} \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

where `extract_match(y)` match the target answer and generated response.

2. Format Reward (r_{num}) Ensures the correctness of the answer structure and valid numerical reasoning. The reward is computed as:

$$r_{ans}(y) = \begin{cases} 0.5 & \text{if } \texttt{answer\_format}(y) \text{ is valid} \\ 0 & \text{otherwise} \end{cases} \quad (11)$$

follow `The answer is: []` (in specific language).

3. Format Reward (r_f) Enforces compliance with a rigid reasoning structure using regex:

$$r_f(y) = \begin{cases} 0.5 & \text{if response matches } s_1 \wedge s_2 \\ 0 & \text{otherwise} \end{cases} \quad (12)$$

Ensures that responses follow explicit structured reasoning process elicited by SWATH.

4. Soft Format Reward (r_s) Provides flexibility and maintains structure by checking for the presence of reasoning and answer tags:

$$r_s(y) = \begin{cases} 0.5 & \text{if response matches } s_1 \ast\wedge s_2 \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

Allows minor deviations and ensures reasoning and answers remain explicitly structured.

5. Structural Integrity Reward (r_{SI}) Assign incremental rewards on correct placement and penalising excessive content:

$$r_{SI}(y) = \sum_{i=1}^{4} w_i \cdot \mathbb{1}(s_i \in y) - \lambda \cdot \textit{extra\_content} \quad (14)$$

where: $w_i = 0.125$ for placing s_1 and s_2 and $\lambda = 0.001$ additional content.

N DPO Tuning

In parallel with the GRPO method (Appendix M), we use DPO as a preference optimisation algorithm. As mentioned in §3, DPO was designed to lower the costs of the reward model and have a stronger binary choice. This algorithm has been used in different configurations. In this paper, we adopt the configuration proposed in (Ranaldi and Freitas, 2024b) by extending the functionality in multilingual contexts and using prompt-based demonstrations of our SWATH.

Starting from the demonstrations defined as $\mathcal{D} = (x_i, a_i)$ where $i \in \mathcal{D}$ (note that a_i are generated using SWATH), we instruct the models using the input x_i $\forall i \in \mathcal{D}$ (the latter being the proposed approach in the main paper). For each element, we collect the **Answers** ($y_i = \pi_\theta(x_i)$), which are the responses generated by the model given the input x_i.

We define the following components:

- **Oracle or Target** t_i: the expected target answer given the input x_i.
- **Demonstration Answer** $\hat{a}_i$ and a_i: target answers given the input x_i or $\hat{x}_i$.
- **Answer** $y_i = \pi_{\text{inst}}(x)$: the response generated by the model given the input x.
- **SWATH Answer** $y_{\text{SWATH}} = \pi_{\text{inst}}(x_{\text{SWATH}})$: the structured response generated via the SWATH mechanism x_{SWATH}.

After an initial warm-up phase using SFT, we apply the DPO, selecting y_w following function:

$$y_w = \begin{cases} \hat{y}_i & \text{if } t_i \in \hat{y}_i \wedge \texttt{form}(\hat{y}_i) \\ \hat{a}_i & \text{otherwise} \end{cases} \quad (15)$$

while the discouraged answers y_l are given by y_i $\forall i \in \mathcal{D}$. The function hasStep($\hat{y}_i$) returns `'True'` when both constraints of the function are satisfied otherwise `'False'`.

Specifically, given an output y:

$$\texttt{form}(y) = \begin{cases} \texttt{True} & \text{if } (s_1 \wedge s_2) \in y \\ \texttt{False} & \text{otherwise} \end{cases} \quad (16)$$

where s_1 and s_2 are respectively "**<formalisation>...</formalisation>**" and "**<reasoning>...</reasoning>**" (SWATH in Table A).