Multilingual Self-Taught Faithfulness Evaluators

Carlo Alfano[1,2], Aymen Al Marjani[2], Zeno Jonke[2],
Amin Mantrach[2], Saab Mansour[2], Marcello Federico[2]
[1]University of Oxford, [2]Amazon
Correspondence: alfancar@amazon.es

Abstract

The growing use of large language models (LLMs) has increased the need for automatic evaluation systems, particularly to address the challenge of information hallucination. Although existing faithfulness evaluation approaches have shown promise, they are predominantly English-focused and often require expensive human-labeled training data for fine-tuning specialized models. As LLMs see increased adoption in multilingual contexts, there is a need for accurate faithfulness evaluators that can operate across languages without extensive labeled data. This paper presents STEMF (Self-Taught Evaluators for Multilingual Faithfulness), a framework that learns exclusively from synthetic multilingual summarization data while leveraging cross-lingual transfer learning. Through experiments comparing language-specific and mixed-language fine-tuning approaches, we demonstrate a consistent relationship between an LLM's general language capabilities and its performance in language-specific evaluation tasks. Our framework shows improvements over existing baselines, including state-of-the-art English evaluators and machine translation-based approaches. Our code is available at https://github.com/amazon-science/multilingual-faithfulness

1 Introduction

The growing application of large language models (LLMs) for text generation tasks across multiple languages has created an urgent need for efficient and reliable evaluation methods. While human evaluation remains the gold standard for assessing LLM outputs, it is expensive, time-consuming, and difficult to scale across different languages. This has led to increasing interest in automated evaluation approaches, particularly using LLMs themselves as evaluators (Li et al., 2023; Zheng et al., 2023).

Recent work has demonstrated promising results using LLMs to evaluate other LLMs' outputs, especially for tasks like summarization and QA in English (Liu et al., 2023; Song et al., 2024; Wang et al., 2024c). However, extending these evaluation approaches to multiple languages presents several key challenges. First, most existing LLM-as-judge frameworks have been developed and tested primarily on English data. Second, it remains unclear whether LLMs can deliver consistent evaluation quality across different languages, especially for languages with limited training data. Third, the optimal approach for training multilingual LLM evaluators - whether through language-specific fine-tuning, mixed-language training, or other strategies - is not well understood. In this paper, we address these challenges by developing a framework for training multilingual faithfulness evaluators with synthetic data, using a self-taught approach (Wang et al., 2024b). Our key contributions include:

- **S**elf-**T**aught **E**valuators for **M**ultilingual **F**aithfulness (**STEMF**), a scalable framework for training multilingual faithfulness evaluators using synthetic summarization data.
- Comprehensive experiments across multiple languages and model architectures to identify optimal training strategies.
- A multilingual finetuned faithfulness evaluator based on a 9B-parameter LLM that outperforms baselines of similar dimension and matches the performance of larger models.
- Comparison to baselines such as out-of-the-box prompting and translation based approaches.

Through extensive experimentation, we demonstrate that our framework can effectively improve evaluation performance across languages while identifying important considerations for deploying such systems in practice. Our findings provide valuable insights for researchers and practitioners working on multilingual LLM evaluation.

The rest of this paper is organized as follows. Section 2 reviews related works and Section 3

Findings of the Association for Computational Linguistics: EACL 2026, pages 5035–5051
March 24-29, 2026

presents the STEMF framework in detail. Section 4 describes our experimental setup, with results and analysis presented in Section 5.

2 Related Work

Our work is related to three main topics present in the literature: LLMs as evaluators, LLMs taught by LLMs, and multilinguality in LLMs.

LLMs as evaluators Given the cost of acquiring human annotations, the automatic and effective evaluation of LLMs is a crucial challenge in LLM research that has been growing in difficulty with the advancement of LLM capabilities. Traditional metrics for the automatic evaluation of LLMs, such as BLEU and ROUGE, have been shown to have poor correlation with human annotations when evaluating the output of modern LLMs for most tasks (Blagec et al., 2022), urging researchers to find innovative solutions. A promising approach to address this challenge consists in using LLMs to evaluate LLMs, either by designing prompts for off-the-shelf instruction tuned models (Li et al., 2023; Zheng et al., 2023; Saha et al., 2024), or by fine-tuning LLMs to act as evaluators (Kim et al., 2024; Wang et al., 2024a; Tang et al., 2024a), especially in a self-supervised fashion.

LLMs taught by LLMs The cost of human annotations affects the fine-tuning of LLMs as well, and researcher have devised methods to fine-tune LLMs using LLMs (Lakew et al., 2017; Pan et al., 2024). Yuan et al. (2024) showed that it is possible to fine-tune an LLM using rewards given by the LLM itself. Building on the same idea, (Wu et al., 2024) showed that allowing the LLM to evaluate its own judgments and using the feedback to improve its judgments skills leads to larger improvements than the original strategy. As to evaluators, Wang et al. (2024b) and (Tang et al., 2024a) have built LLM-based evaluators using contrastive synthetic annotated data. In particular, they have shown that constructing a dataset by asking an LLM to generate good and bad responses to a set of prompts, and then fine-tuning an LLM on the synthetic dataset, leads to state-of-the-art evaluators. Most existing works on this topic focus on the English language, highlighting a gap in the literature regarding the evaluation of multilingual LLMs evaluators, which we plan to fill.

Multilinguality in LLMs In terms of multilingual LLMs, designing training strategies that allow generalization or transfer to different languages is crucial to address the lack of high-quality data in many languages. Artetxe et al. (2020) and later Chen et al. (2023) propose to transfer an LLM to a new language by freezing all the parameters of the LLM except the embedding, which is reinitialized and re-learned through pretraining on the target language. Zhao et al. (2024a) show that, if provided with sufficient data (1B tokens), these strategies are not necessary and standard pretraining and fine-tuning are sufficient. Bacciu et al. (2024) show that performing SFT with QLora on Mistral 7B-Instruct-v0.2 on several Italian datasets increases the performance of the LLM on Italian benchmarks. Besides training, recent works (Wendler et al., 2024; Tang et al., 2024c; Zhao et al., 2024b) have studied how LLMs handle multilingualism and observed that internal layers of the LLM deal with abstract concepts and task solving, while external layers address the input and output languages.

Finally, we note that Bavaresco et al. (2024) and Tang et al. (2024b) provide evidence that LLMs are not yet ready to systematically replace human judgments, as their agreement with human preferences varies largely across tasks and is particularly lacking on dialogue data.

3 STEMF Framework

Our objective is to build a model that, given a document and a paragraph, returns a prediction of the faithfulness of the paragraph to the document. Our implementation is based on FineSurE (Song et al., 2024), which consists of providing a prompt to an instruction-tuned LLM asking it to judge the faithfulness of a summary to the original document sentence-wise, based on a rubric of faithfulness error definitions. The LLM is also asked to specify what type of faithfulness error the sentence contains and provide an explanation for its judgment. We choose this implementation because it has shown state-of-the-art performance among prompt-based methods, and it has a simple implementation.

To further simplify the evaluator implementation, instead of feeding the model the entire paragraph, we split the paragraph into sentences and ask the model to judge the faithfulness of each sentence separately. We find that this modification helps our methodology. The prompt provided to the LLM is outlined in Listing 1 in Appendix B.

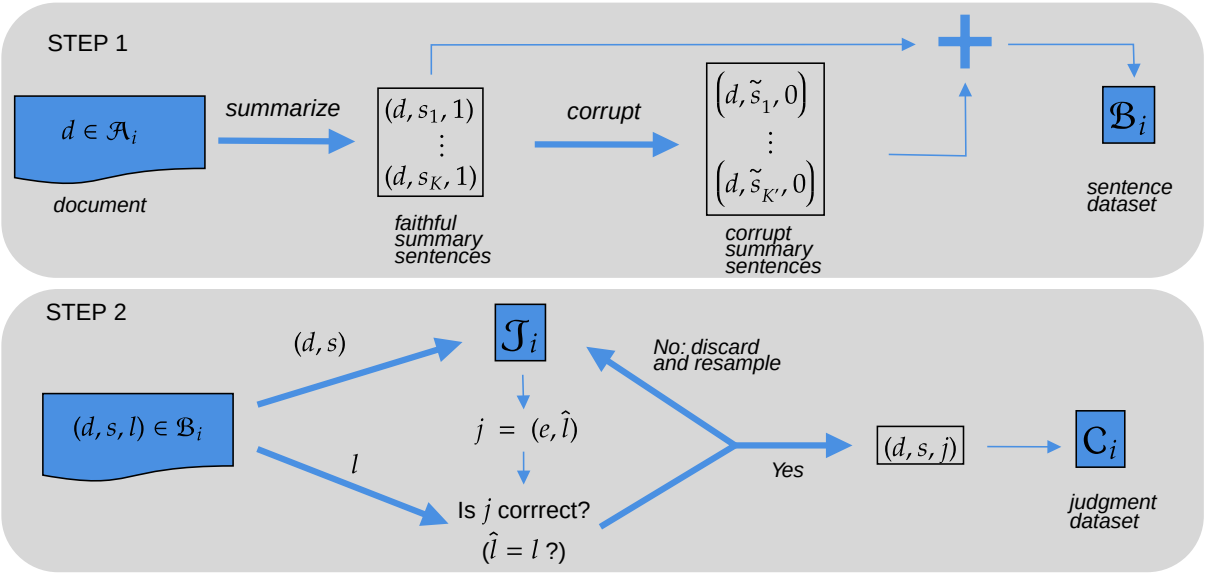

Figure 1: Our framework for self-taught multilingual evaluators STEMF includes two main steps: summary generation with/out corruption (Step 1) and judgments generation (Step 2). The judgment $j = (e, \widehat{l})$, which consists of an explanation e and a faithfulness prediction $\widehat{l}$, is sampled from the current evaluator $\mathcal{J}_i$ given the document-sentence pair (d, s). If the faithfulness prediction $\widehat{l}$ agrees with the label l, the judgment is accepted and added to the judgment dataset $\mathcal{C}_i$ with the associated document-sentence pair.

3.1 Methodology

The STEMF approach builds a self-taught faithfulness judge in four steps, which are performed sequentially and repeated for several iterations. We provide a visual description in Figure 1. Denote by $\mathcal{J}_1$ the starting LLM that will be tuned through our methodology, by $\mathcal{J}_{i+1}$ the LLM obtained after iteration i, and by $\mathcal{M}$ an auxiliary LLM. Let Ω be a multilingual document dataset. Our goal is to generate training data points of the form (d, s, j), where $j = (e, \widehat{l})$ is a judgment on the faithfulness of the sentence s to the document $d \in \Omega$ and consists of an explanation e and a faithfulness prediction $\widehat{l}$. All the prompts given to $\mathcal{J}_i$ and $\mathcal{M}$ are provided in Appendix B. For each iteration $i \geq 1$, we perform the following steps.

Step 0. Document selection. We build the document dataset $\mathcal{A}_i$ by randomly sampling one thousand documents from Ω, equally distributed among selected languages.

Step 1. Summary generation. For each document $d \in \mathcal{A}_i$, we generate a (pseudo-)faithful and a (pseudo-)corrupt summary, which are split into sentences and added to the sentence dataset $\mathcal{B}_i$. We consider two strategies, which are summarized in Figure 4 in Appendix C and explained below.

- The first strategy consists in asking $\mathcal{M}$ to produce a faithful summary for the document and a corrupt version of the document $\widetilde{d}$, which should contain information contradicting the original document in each of its sentences. A summary for the corrupt document is obtained by asking $\mathcal{M}$ to produce a faithful summary. We call this the *indirect* corruption strategy.
- The second strategy consists in asking $\mathcal{M}$ to produce a faithful summary consisting of sentences $(s_1, ..., s_K)$. We then obtain corrupt summary sentences by asking $\mathcal{M}$ to introduce a specific faithfulness error in each sentence of the faithful summary (s_k is corrupted into $\widetilde{s}_k$). We call this the *direct* corruption strategy.

Both strategies produce the document-sentence-label triplets $\{(d, s_k, 1)\}_{k=1}^{K}$ and $\{(d, \widetilde{s}_k, 0)\}_{k=1}^{K'}$, which are added to the sentence dataset $\mathcal{B}_i$.

Step 2. Judgment generation. For each document-sentence-label triplet in $\mathcal{B}_i$, we sample a judgment j from $\mathcal{J}_i$, as summarized in Figure 1. A judgment consists of faithfulness *prediction* $\widehat{l}$, i.e., whether the sentence s is faithful to the document d, and an *explanation* e of the prediction. If the prediction is correct according to the pseudo-labels generated in Step 1, the document-sentence-judgment triplet (d, s, j) is added to the judgment dataset $\mathcal{C}_i$. If the judgment is incorrect, a new judg-

ment is sampled and the same process is repeated. We sample a new judgment up to two times.

Step 3. Fine-tuning The last step is to fine-tune $\mathcal{J}_i$ on the judgment datasets $\mathcal{C}_i$, obtaining $\mathcal{J}_{i+1}$. For this step, we used supervised fine-tuning (SFT) with low rank adapters (LoRA) (Hu et al., 2022).

When $\mathcal{J}_1$ is initialized with $\mathcal{M}$, the final evaluator J_r (assuming r iterations) is called *self-taught*, as it is trained on data produced by itself. In practice, since $\mathcal{M}$ remains static throughout the procedure, we generate all faithful summaries, corrupt documents and corrupt summaries once and take one thousand samples at each iteration. The algorithm is summarized in Appendix C.

3.2 Variations

Besides the above **base** methodology, we propose three fine-tuning variations to inform practitioners on which strategies are effective in improving the final performance.

1. **Only central layers.** The first variation modifies the *fine-tuning* step and consists in freezing the first and last 25% of the layers of the LLM and only training the central layers. Following the intuition from Wendler et al. (2024) and Tang et al. (2024c), we posit that training only central layers of an LLM improves the LLM's capability at solving a particular task, without impacting its multilingual understanding and generation capabilities.

2. **Proxy data.** Our second variation explores the effectiveness of using related data, natural language inference (NLI) data in our case, to the dataset used in the *fine-tuning* step (Step 3). We include 20k examples from the multilingual dataset XNLI (Conneau et al., 2018), formatting them as described in Appendix B.3.

3. **Human labels.** Lastly, we consider the setting where half of the document-sentence-label triplets in the sentence dataset $\mathcal{B}_i$ have been labeled faithful or unfaithful by a human. This addresses the issue of judgment quality assurance, as we cannot guarantee the correctness of faithful summaries or the unfaithfulness of corrupt summaries when they are generated by $\mathcal{M}$ in the base methodology.

4 Experimental Setup

In this section, we describe our experimental setup. We start by discussing datasets and benchmarks, then follow by outlining our experiments.

Datasets for training For all self-supervised experiments, Ω is the wikilingua dataset (Ladhak et al., 2020), which is a collection of *wikihow* articles from multiple languages. The languages we consider in our investigation are German, English, Spanish, French, Hindi, Arabic, and Italian. To understand the impact of supervision and implement the third variation to the base methodology, we use human labeled data from the FRANK dataset (Pagnoni et al., 2021), which contains English articles from CNN/DM (Hermann et al., 2015) and XSum (Narayan et al., 2018) together with LLM-generated summaries annotated sentence-wise for faithfulness. We machine translate FRANK to all the other languages we are considering, apart from Hindi since we found, through manual inspection, the translation to have low quality.

Evaluation We evaluate the trained LLMs on summarization and retrieval-augmented generation (RAG) benchmarks, using the balanced accuracy metric, whose definition is recalled in Appendix D. The first two benchmarks are the MEMERAG (Cruz Blandón et al., 2025) and the mFACE (Aharoni et al., 2023) datasets, which contain native multilingual faithfulness data. MEMERAG is a dataset of question-answer-label triplets, where the answer is generated by an LLM using supporting wiki articles and is labeled by a human according to its faithfulness to the supporting articles. mFACE contains articles-summary-label triplets, where the article is taken from XL-sum (Hasan et al., 2021), the summary is generated by an LLM, and the label says whether the summary is faithful to the article or not. We also test on the translated FRANK dataset. For details on each benchmark, see Table 10 in the Appendix.

Experiments We perform extensive testing of our methodology. Firstly, we explore the impact of the languages included in $\mathcal{A}_i$ and of the starting model $\mathcal{J}_1$, when using the indirect corruption strategy and setting $\mathcal{M}$ as the starting LLM $\mathcal{J}_1$. We trained each starting model on each of the first five considered languages, on a mix of all seven languages (*all*), and on a mix of the five European languages (*euro5*). The starting models are:

	English	French	German	Hindi	Spanish	Arabic	Italian	English	
	MEMERAG					mFACE	FRANK		Avg
Models trained with our method STEMF									
Qwen-2.5-7B-Instruct	69.6	63.5	57.9	57.5	61.6	64.2	68.4	75.0	65.1
+ STEMF	77.8	71.5	69.5	<u>81.3</u>	<u>77.7</u>	72.7	82.6	86.0	77.4
gemma-2-9b-it	76.3	72.1	67.9	69.1	73.8	72.8	81.5	85.5	75.6
+ STEMF	78.4	<u>74.3</u>	70.6	**81.4**	77.0	72.0	<u>85.0</u>	<u>86.8</u>	<u>78.5</u>
SOTA									
MiniCheck-7B	<u>79.1</u>	66.1	<u>71.5</u>	73.0	73.2	<u>73.4</u>	81.6	84.6	76.1
Qwen2.5-72B-Instruct	**79.6**	**75.6**	**76.1**	76.7	**81.0**	**76.7**	**85.6**	**87.5**	**80.0**

Table 1: Balanced accuracy (in percentage) of our best trained evaluators, compared to the starting LLMs and strong baselines. The evaluations on English, French, German, Hindi, and Spanish are done on the MEMERAG dataset, while the evaluations for Arabic and Italian are done on the mFACE and FRANK datasets, respectively. We also report the performance on the original English FRANK. The Avg column reports the balanced accuracy averaged over all datasets and languages. We report in bold and underlined the highest and second highest values, respectively, for each column.

(i) Qwen2.5-3B-Instruct, Qwen2.5-7B-Instruct[*], (ii) gemma-2-2b-it, gemma-2-9b-it[†], (iii) Mistral-Nemo-Instruct-2407[‡]. We then test the direct corruption strategy and the three variations proposed in Section 3.1. For these last experiments, we use different combinations of training languages and the `Qwen2.5-7B-Instruct` model, which was the best performing model that fit in our computational setup. Lastly, a common and simple approach to tackle multilinguality is machine translating (MT) target languages to English and running inference using the English MT data, eg (Artetxe et al., 2023), which we test here. Implementation details and hyperparameters are reported in Appendix E.

5 Results

We provide a discussion of our results. We start by comparing the performance of our best trained evaluators against strong baselines, namely `Bespoke-MiniCheck-7B` (Labs, 2024), a SOTA faithfulness evaluator trained on English data only, and `Qwen2.5-72B-Instruct`, a best in its class on multilingual tasks (Qwen et al., 2025)[§]. Later in this section we discuss the results of the experiments outlined in Section 4.

Our best trained evaluators are `gemma-2-9b-it` and `Qwen2.5-7B-Instruct`, trained with the base STEMF framework, on exclusively English data, with the indirect corruption strategy. Their balanced accuracy is reported in Table 1 against baselines and the starting models. Both trained models outperform `Bespoke-MiniCheck-7B`, in terms of average balanced accuracy, and achieve results closer to those of `Qwen2.5-72B-Instruct`, a model eight times larger than `gemma-2-9b-it`. We also note that our trained version of `gemma-2-9b-it` has a sentence-wise balanced accuracy of 87.7% on FRANK (English), against 86.4% reported for GPT-4 by Song et al. (2024) and 87.5% achieved by `Qwen2.5-72B-Instruct`. To ensure these results are reproducible, we perform two additional training runs for `gemma-2-9b-it` and `Qwen2.5-7B-Instruct` and report the results Table 9 in Appendix F.

5.1 Impact of language

We investigate the impact of training languages on evaluators trained with the base `STEMF`framework and the indirect corruption strategy. We report the improvement in balanced accuracy over the starting model of each trained evaluator, averaged over all test languages and benchmarks, in Figure 2. The exact improvement and final balanced accuracy are provided in the Appendix in Tables 12 and Table 13, respectively. Our first observation is that training on exclusively English data leads to better results, on average, than any other combination of languages. As shown in Figure 2, evaluators trained on English present a 7.8% average increase

[*]https://huggingface.co/Qwen
[†]https://huggingface.co/google
[‡]https://huggingface.co/mistralai
[§]At the time of this work development, as compared to similar sized models - Llama3.1-70B, Mistral-Large, GPT4o-mini. https://huggingface.co/Qwen.

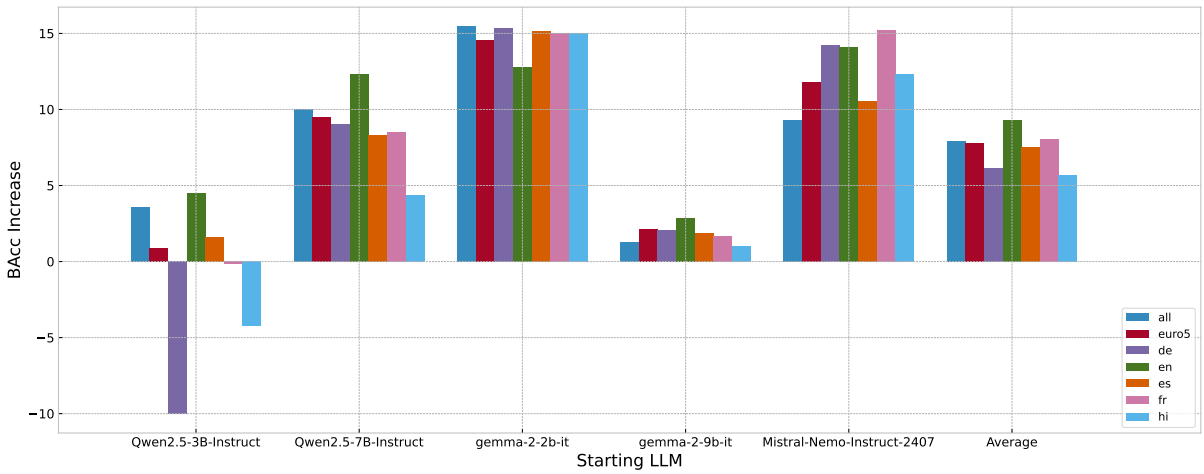

Figure 2: A comparison of training languages combinations across various LLMs. The plot reports the *improvement in balanced accuracy* over the base model for each language combination used in training, in percentage points. The balanced accuracy (y-axis) is averaged over all benchmarks and languages. The models are trained using the base methodology and the indirect corruption strategy.

in balanced accuracy, while the second highest average improvement is 6.2% and is for evaluators trained on the *all* and *euro5* language mixes. The lowest average improvement is 4.4% and is for evaluators trained on Hindi. We also observe that there is no significant difference in average balanced accuracy increase (p-value > 0.1) between training on a mixture of languages and training on a single language. Aside from the impact of language, these experiments also show that weaker models, such as `gemma-2-2b`, benefit largely from our training framework while the improvements are lower for the stronger counterpart, i.e. `gemma-2-9b-it`.

These results, together with the observations on the performance of the starting models made in the previous section, suggest that **the improvements brought by our methodology are proportional to the performance of the starting model in the training languages**. We confirm this behaviour in Figure 3, where we plot for some of our trained models the increase in average balanced accuracy against the balanced accuracy of the starting model and the MMMLU[¶] scores of the starting model, both averaged across the languages used in training. The MMMLU scores of all starting models are reported in Table 11 in the Appendix. We find that **including the test language among the training languages is beneficial only if the starting model is proficient enough on that test language**. We see a correlation of 0.20 (p-value $<$ 0.01) between the balanced accuracy of the trained evaluator and the presence of the test language in the training set, provided the MMMLU score of the starting model on the test language is higher than 65%. In contrast, the correlation between the balanced accuracy of the trained evaluator and the presence of the test language in the training set, provided the MMMLU score of the starting model on the test language is lower than 65%, is -0.21 (p-value $<$ 0.01). Similarly, the correlation between the increase in balanced accuracy of the trained evaluator and the presence of the test language in the training set, provided the MMMLU score of the starting model on the test language is lower than 65%, is -0.18 (p-value $<$ 0.01).

Next, we investigate the impact of the number of training iterations on the final model. We report in Table 8 in the Appendix the average improvement in balanced accuracy for all models trained on English, at different iterations. We observe that **the third and fifth iterations are better than the first, but there is no clear difference between running the methodology for 3 or 5 iterations**.

Our investigation reveals that the set of languages included in the document dataset $\mathcal{A}_i$ has a significant influence on the final performance of the trained evaluator. In particular, our takeaway from this first batch of experiments is that it is important to train the starting model only on languages that it is fluent in (as measured by the MMMLU performance), regardless of the test language. Moreover, we observe that training on only English data is a good choice.

[¶]https://huggingface.co/datasets/openai/MMMLU, human translation of the MMLU dataset.

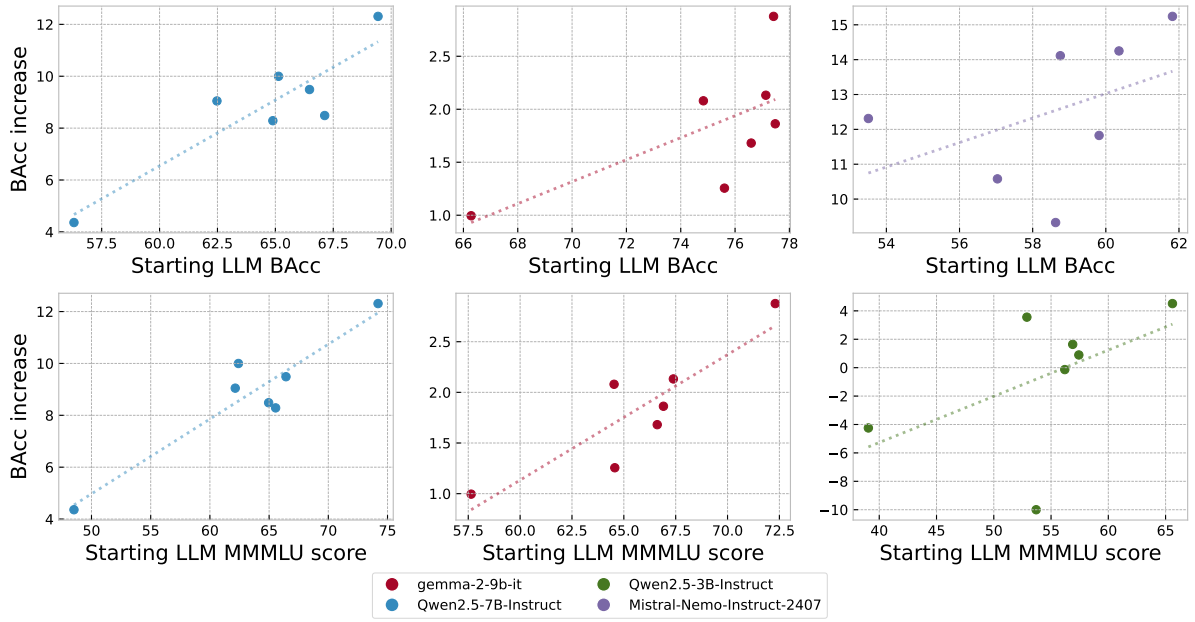

Figure 3: Relationship between performance improvement and performance of the starting LLM across the different training languages. For each trained model (represented as a dot), the plots report the improvement in averaged balanced accuracy over the starting model (y-axis) against the balanced accuracy (top x-axis) and the MMMLU score (bottom x-axis) of the starting model, averaged over the languages used in training. Each plot is specific to an LLM and present a regression line of the balanced accuracy increase against either the starting LLM balanced accuracy (top) or the starting LLM MMMLU score (bottom).

5.2 Corruption strategy

$\mathcal{M}$	Qwen-72B	Qwen-7B	
Corr. strat.	direct	direct	indirect
all	9.5	3.9	10.0
euro5	9.4	7.6	9.5
en	10.0	10.2	12.3
Average	9.6	7.2	10.6

Table 2: Improvement in balanced accuracy for Indirect vs direct data synthesis strategies. We use `Qwen2.5-7B-Instruct` as the base model, and try different combinations of training languages and auxiliary model $\mathcal{M}$. The balanced accuracy is averaged over all languages and benchmarks.

To test which corruption strategy is more effective, we trained the `Qwen2.5-7B-Instruct` model with the direct and indirect corruption strategies. Regarding the languages used in training, we consider the *all*, *euro5* and English only combinations. We initially noted that `Qwen2.5-7B-Instruct` struggles inserting faithfulness mistakes to generate corrupted summary sentences, hence we added an experiment for the direct corruption method using `Qwen2.5-72B-Instruct`. The improvement in averaged balanced accuracy over the starting model is reported in Table 2. When using `Qwen2.5-7B-Instruct` as $\mathcal{M}$, we see that the direct corruption strategy is less effective than the indirect one, especially when including languages different from English. When using `Qwen2.5-72B-Instruct` as $\mathcal{M}$ for the direct strategy, the difference between the performance of the two strategies gets smaller, but the indirect strategy is still uniformly better. We conclude that **the indirect corruption strategy is simpler to execute and can be performed by a relatively smaller LLM. On the contrary, the direct corruption strategy is more complex, especially in a multilingual context, and requires a large model with good multilingual capabilities**.

5.3 Variations

Training Central Layers We trained the central layers of the `Qwen2.5-7B-Instruct` model for three different combinations of languages used in training and compared the obtained evaluators with fully trained models on the same language combinations. We report the results in Table 3, which shows that, after five iterations, there is no significant difference between the two strategies. However, the performance of the evaluators for

	Iteration 1		Iteration 5	
	Full model	Only central layers	Full model	Only central layers
all	3.4 (0.4)	5.6 (6.5)	10.0 (10.5)	9.6 (10.8)
en-it	1.9 (1.4)	2.9 (3.8)	9.9 (11.4)	11.6 (11.7)
en	7.8 (7.2)	7.5 (7.9)	12.3 (13.6)	12.1 (12.7)
Average	4.4 (3.0)	5.4 (6.1)	10.7 (11.8)	11.1 (11.7)

Table 3: Comparison of full model training against training only central layers, at iterations 1 and 5. The table reports the improvement in balanced accuracy averaged over all languages and all benchmarks (averaged only over the MEMERAG dataset), for different combinations of languages used in training.

which we only trained the central layers is higher in the first iteration, especially on the MEMERAG dataset. This behavior suggests that **training only the central layers leads to a faster convergence of `STEMF`**. Regarding computational savings, initializing and training LoRA adapters only for central layers requires half the memory of doing the same for the whole model. However, the costs of keeping the frozen full model in memory, as well as the inference costs once the adapters are merged with the full model, stay the same. Given our results and the reduced computational cost of training only the central layers of an LLM, we recommend doing so in practice.

Proxy data We report the performance of the STEMF framework with proxy data, compared to the performance of the base framework, in Table 7 in the Appendix. Our approach to including XNLI data in the training dataset did not bring improvements, on the contrary, it hurts performance in two out of three language combinations we tried. This is in contrast to previous observations, as Tang et al. (2024a) found that NLI was helpful to their methodology to train a binary classifier.

Human labels We replace 50% of the synthetic pseudo labeled data generated in Step 1 (sentence dataset $\mathcal{B}_i$) with human-labeled data for faithfulness from the FRANK dataset. We perform the experiment to quantify the quality of our pseudo-labels and whether human labels are needed. We find that training with human-labeled data on the *all* language combination leads to an improvement in balanced accuracy of 8.0%, against a 10% improvement obtained by the base methodology relying on pseudo-labels only. This result indicates that synthetic data generated from the `wikilingua` dataset is sufficient for our purposes, and seems to enable better generalization of the trained model to the other datasets, i.e. MEMERAG and mFACE.

5.4 Baselines with Translation

To understand whether translation can help in evaluating faithfulness, we re-evaluated some of the starting models on the MEMERAG dataset, but translating everything to English. The change in balanced accuracy is reported in Table 4 which shows that translation often hurts performance, with the exception of Hindi where translation brings improvements almost everywhere, probably due to the low proficiency of the models in this language.

	de	es	fr	hi
Mistral-Nemo-Instruct	-1.6	2.6	0.5	0.7
Qwen2.5-3B-Instruct	-2.5	-1.5	-5.2	4.1
Qwen2.5-7B-Instruct	0.3	-3.0	-3.3	2.6
gemma-2-9b-it	-2.6	-5.4	-1.8	1.4

Table 4: Evaluators' results using the target language vs. a "pivot" approach of translating into English first. We report the difference in balanced accuracy, for several starting models, on the MEMERAG dataset.

6 Conclusion

We presented `STEMF`, a framework to train LLM-based multilingual faithfulness evaluators through synthetic data, which led to substantial improvements across most starting LLMs, with an average increase of 6.9% in balanced accuracy. Our best evaluator, based on `gemma-2-9b-it` and trained on English data, achieves competitive performance with much larger models and even outperforms GPT-4 on the FRANK benchmark. We identified several key insights for developing effective evaluators across languages. Our results indicate that practitioners should prioritize selecting starting models with strong proficiency in their target languages when performing self-taught approaches, with training on English often yielding the best results across all test languages.

Additionally, we found that training only the central layers of an LLM achieves comparable results to training all layers while being more computationally efficient. Lastly, we found that translation-based approaches generally degrade performance except for languages where the LLM has a very limited proficiency.

Limitations

Our STEMF methodology is primarily focused on faithfulness evaluation. In practice however, there exists several other dimensions that are required to get a comprehensive evaluation of LLM outputs in the settings of summarization and QA, e.g. informativeness, harmfulness and fluency. We leave this investigation for future work. Under the experimental conditions herein described, our results show that training on English is sufficient. We hypothesize that this relates to strong general language capability on English in comparison to other languages, and a language agnostic representation of faithfulness features, which is left for future work. Finally, our study would not have been possible without the faithfulness benchmarks covering the selected languages. We believe that it is crucial to extend these benchmarks, as well as the automatic evaluation research, to low-resource languages in order to ensure the usefulness and safety of LLM-generated content across all cultures.

References

Roee Aharoni, Shashi Narayan, Joshua Maynez, Jonathan Herzig, Elizabeth Clark, and Mirella Lapata. 2023. Multilingual summarization with factual consistency evaluation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3562–3591, Toronto, Canada. Association for Computational Linguistics.

Mikel Artetxe, Vedanuj Goswami, Shruti Bhosale, Angela Fan, and Luke Zettlemoyer. 2023. Revisiting machine translation for cross-lingual classification. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6489–6499, Singapore. Association for Computational Linguistics.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Andrea Bacciu, Cesare Campagnano, Giovanni Trappolini, and Fabrizio Silvestri. 2024. DanteLLM: Let's push Italian LLM research forward! In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4343–4355, Torino, Italia. ELRA and ICCL.

Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, et al. 2024. Llms instead of human judges? a large scale empirical study across 20 nlp evaluation tasks. *CoRR*.

Kathrin Blagec, Georg Dorffner, Milad Moradi, Simon Ott, and Matthias Samwald. 2022. A global analysis of metrics used for measuring performance in natural language processing. In *Proceedings of NLP Power! The First Workshop on Efficient Benchmarking in NLP*, pages 52–63, Dublin, Ireland. Association for Computational Linguistics.

Yihong Chen, Kelly Marchisio, Roberta Raileanu, David Adelani, Pontus Lars Erik Saito Stenetorp, Sebastian Riedel, and Mikel Artetxe. 2023. Improving language plasticity via pretraining with active forgetting. *Advances in Neural Information Processing Systems*, 36:31543–31557.

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

María Andrea Cruz Blandón, Jayasimha Talur, Bruno Charron, Dong Liu, Saab Mansour, and Marcello Federico. 2025. Memerag: A multilingual end-to-end meta-evaluation benchmark for retrieval augmented generation. *Preprint*, arXiv:2502.17163.

Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. 2021. XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703, Online. Association for Computational Linguistics.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and

Minjoon Seo. 2024. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Bespoke Labs. 2024. Bespoke-minicheck-7b.

Faisal Ladhak, Esin Durmus, Claire Cardie, and Kathleen McKeown. 2020. WikiLingua: A new benchmark dataset for cross-lingual abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4034–4048, Online. Association for Computational Linguistics.

Surafel M Lakew, Quintino F Lotito, Matteo Negri, Marco Turchi, and Marcello Federico. 2017. Improving zero-shot translation of low-resource languages. In *Proceedings of the 14th International Conference on Spoken Language Translation*, pages 113–119.

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

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2024. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. *Transactions of the Association for Computational Linguistics*, 12:484–506.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Swarnadeep Saha, Omer Levy, Asli Celikyilmaz, Mohit Bansal, Jason Weston, and Xian Li. 2024. Branch-solve-merge improves large language model evaluation and generation. 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 8352–8370, Mexico City, Mexico. Association for Computational Linguistics.

Hwanjun Song, Hang Su, Igor Shalyminov, Jason Cai, and Saab Mansour. 2024. FineSurE: Fine-grained summarization evaluation using LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 906–922, Bangkok, Thailand. Association for Computational Linguistics.

Liyan Tang, Philippe Laban, and Greg Durrett. 2024a. MiniCheck: Efficient fact-checking of LLMs on grounding documents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8818–8847, Miami, Florida, USA. Association for Computational Linguistics.

Liyan Tang, Igor Shalyminov, Amy Wong, Jon Burnsky, Jake Vincent, Yu'an Yang, Siffi Singh, Song Feng, Hwanjun Song, Hang Su, Lijia Sun, Yi Zhang, Saab Mansour, and Kathleen McKeown. 2024b. TofuEval: Evaluating hallucinations of LLMs on topic-focused dialogue summarization. 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 4455–4480, Mexico City, Mexico. Association for Computational Linguistics.

Tianyi Tang, Wenyang Luo, Haoyang Huang, Dongdong Zhang, Xiaolei Wang, Xin Zhao, Furu Wei, and Ji-Rong Wen. 2024c. Language-specific neurons: The key to multilingual capabilities in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5701–5715, Bangkok, Thailand. Association for Computational Linguistics.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Shengyi Huang, Kashif Rasul, Alvaro Bartolome, Alexander M. Rush, and Thomas Wolf. The Alignment Handbook.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024a. Interpretable preferences

via multi-objective reward modeling and mixture-of-experts. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10582–10592, Miami, Florida, USA. Association for Computational Linguistics.

Tianlu Wang, Ilia Kulikov, Olga Golovneva, Ping Yu, Weizhe Yuan, Jane Dwivedi-Yu, Richard Yuanzhe Pang, Maryam Fazel-Zarandi, Jason Weston, and Xian Li. 2024b. Self-taught evaluators. *arXiv preprint arXiv:2408.02666*.

Yidong Wang, Zhuohao Yu, Wenjin Yao, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, Wei Ye, Shikun Zhang, and Yue Zhang. 2024c. PandaLM: An automatic evaluation benchmark for LLM instruction tuning optimization. In *The Twelfth International Conference on Learning Representations*.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. Do llamas work in English? on the latent language of multilingual transformers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15366–15394, Bangkok, Thailand. Association for Computational Linguistics.

Tianhao Wu, Weizhe Yuan, Olga Golovneva, Jing Xu, Yuandong Tian, Jiantao Jiao, Jason Weston, and Sainbayar Sukhbaatar. 2024. Meta-rewarding language models: Self-improving alignment with llm-as-a-meta-judge. *arXiv preprint arXiv:2407.19594*.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E Weston. 2024. Self-rewarding language models. In *Forty-first International Conference on Machine Learning*.

Jun Zhao, Zhihao Zhang, Luhui Gao, Qi Zhang, Tao Gui, and Xuanjing Huang. 2024a. Llama beyond english: An empirical study on language capability transfer. *arXiv preprint arXiv:2401.01055*.

Yiran Zhao, Wenxuan Zhang, Guizhen Chen, Kenji Kawaguchi, and Lidong Bing. 2024b. How do large language models handle multilingualism? *arXiv preprint arXiv:2402.18815*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

A Evaluators built with synthetic data

In this section, we compare our STEMF framework to those designed by Wang et al. (2024b) and Tang et al. (2024a), who focused exclusively on English.

Wang et al. (2024b) build a self-taught evaluator which, given a generic instruction and two responses, ranks the two responses after providing an explanation. They follow a procedure similar to that outlined in Figure 1, but provide both the good (faithful) and the bad (corrupt) response to the judge and only accept the judgment if it ranks the good response first. In the context of this work, the faithfulness of a summary is a binary variable, which is why we design STEMF to evaluate single summaries and avoid comparisons.

Tang et al. (2024a) build a binary classifier which, given a document and a claim, judges whether the claim is faithful to the document but does not provide an explanation. Similarly to us, they build synthetic faithful and corrupt data, but use a different approach. They produce synthetic data in two ways. The first (Claim to Doc) starts with a claim, decomposes it into atomic facts, and then uses LLMs to generate both supporting and non-supporting documents. The second (Doc to Claim) begins with human-written documents, summarizes them into claims, and then augments this data by modifying/clipping documents or pairing claims with different document sections. Differently from us, their judgment dataset does not include explanations, which could instead help in interpreting predictions.

B Prompts

B.1 Evaluator prompt

The prompt used by our evaluators is provided in Listing 1.

Listing 1: Prompt for the faithfulness evaluator

```
You will receive a text followed by a
    statement. Your task is to assess
    the factuality of the statement with
     respect to the source text across
    nine categories:
* no error: the statement aligns
    explicitly with the content of the
    text and is faithful to it.
* out-of-context error: the statement
    contains information not present in
    the text.
* entity error: the primary arguments (
    or their attributes) of the
    predicate are wrong.
* predicate error: the predicate in the
    statement is inconsistent with the
    text.
* circumstantial error: the additional
    information (like location or time)
    specifying the circumstance around a
     predicate is wrong.
* grammatical error: the grammar of the
    statement is so wrong that it
    becomes meaningless.
* coreference error: a pronoun or
    reference with wrong or non-existing
     antecedent.
* linking error: error in how multiple
    statements are linked together in
    the discourse (for example temporal
    ordering or causal link).
* other error: the statement contains
    any factuality error which is not
    defined here.

Instruction:
First, compare the statement with the
    text.
Second, provide a single sentence
    explaining which factuality error
    the statement has.
Third, answer the classified error
    category for the statement.

Provide your answer in JSON format. The
    answer should be a dictionary whose
    keys are "reason", and "category":
{"reason": "your reason", "category": "
    no error"} or {"reason": "your
    reason", "category": "which error"}

Text:
<replace text here>
Statement:
<replace statement here>
```

B.2 Summary generation prompts

The prompt used to generate a faithful summary of a given article is given in Listing 2. The prompts used to corrupt each sentence in a summary according to the direct corruption strategy depend on the selected type of faithfulness error and are provided in Listing 3 (predicate error), Listing 4 (entity error), Listing 5 (circumstantial error), Listing 6 (linking error), Listing 7 (out-of-context error). Lastly, the prompt used to ask for a corrupted version of an article is given in Listing 8.

Listing 2: Prompt to ask for a faithful summary

```
You will be provided with an article
    containing instructions to complete
    a task or deal with a situation.
    Please provide a concise and
    faithful summary for the article.
    Provide the summary as a list of
    sentences separated by the
    characters '###'. That is,

### First sentence.
### Second sentence.
```

```
### Third sentence.
and so on.

Article:
<replace article here>
```

Listing 3: Prompt to insert a predicate error in a sentence

```
You will be provided with a sentence and
     a source text. First, individuate
    the main clause in the sentence.
    Then, individuate the subject, the
    predicate, the object and the
    attributes of the main clause. Your
    task is to modify the predicate and/
    or the object of the main clause so
    that it is inconsistent with the
    orignal one and the source text.
    Keep the subject and the attributes
    similar to the original sentence.
    Provide your answer in the following
     format.

### Original sentence: <original
    sentence>
### Main clause: <main clause>
### Subject: <subject>
### Predicate: <predicate>
### Object: <object>
### Attributes: <attributes>
### Strategy: <how you are going to
    modify the sentence>
### <modified sentence>

Do not provide additional text after the
     modified sentence.

Sentence:
<replace sentence here>

Source text:
<replace text here>
```

Listing 4: Prompt to insert an entity error in a sentence

```
You will be provided with a sentence and
     a source text. First, individuate
    the main clause in the sentence.
    Then, individuate the subject, the
    predicate, the object and the
    attributes of the main clause. Your
    task is to modify the subject of the
     main clause so that it is
    inconsistent with the original one
    and the text. Keep the predicate,
    the object and the attributes
    similar to the original sentence.
    Provide your answer in the following
     format.

### Original sentence: <original
    sentence>
### Main clause: <main clause>
### Subject: <subject>
### Predicate: <predicate>
### Object: <object>
### Attributes: <attributes>
### Strategy: <how you are going to
    modify the sentence>
### <modified sentence>

Do not provide additional text after the
     modified sentence.

Sentence:
<replace sentence here>

Source text:
<replace text here>
```

Listing 5: Prompt to insert a circumstantial error in a sentence

```
You will be provided with a sentence and
     a source text. First, individuate
    the main clause in the sentence.
    Then, individuate the subject, the
    predicate, the object and the
    attributes of the main clause. Your
    task is to modify the attributes (e.
    g. location, time, manner, direction
    , modality) of the main clause so
    that it is inconsistent with the
    original one and the text. Keep the
    subject, the predicate, and the
    object similar to the original
    sentence. Provide your answer in the
     following format.

### Original sentence: <original
    sentence>
### Main clause: <main clause>
### Subject: <subject>
### Predicate: <predicate>
### Object: <object>
### Attributes: <attributes>
### Strategy: <how you are going to
    modify the sentence>
### <modified sentence>

Do not provide additional text after the
     modified sentence.

Sentence:
<replace sentence here>

Source text:
<replace text here>
```

Listing 6: Prompt to insert a linking error in a sentence

```
You will be provided with a sentence and
     a source text. First, analyze the
    sencence and individuate its clauses
    . Then, modify the sentence so that
    the temporal ordering or the
    discourse links (e.g. RST relations,
     discourse connectors) among its
    clauses are inconsistent with the
    original sentence and the text.
    Provide your answer in the following
     format.

### Original sentence: <original
    sentence>
### First clause: <first clause>
### Second clause: <second clause>
### ...
```

```
### Strategy: <how you are going to
    modify the sentence>
### <modified sentence>

Do not provide additional text after the
     modified sentence.

Sentence:
<replace sentence here>

Source text:
<replace text here>
```

Listing 7: Prompt to insert an out-of-context error in a sentence

```
You will be provided with a sentence and
     a source text. Your task is to
    modify the sentence so that it
    contains information on a matter not
     discussed in the source text.
    Provide your answer in the following
     format.

### Original sentence: <original
    sentence>
### Strategy: <how you are going to
    modify the sentence>
### <modified sentence>

Do not provide additional text after the
     modified sentence.

Sentence:
<replace sentence here>

Source text:
<replace text here>
```

Listing 8: Prompt to corrupt an article

```
You will be provided with an article
    containing instructions to complete
    a task or deal with a situation. The
     article is titled "{title}". Please
     provide contraddicting instructions
     for the same tasks. Make sure the
    new instrcutions are self-coherent
    and plausible. Maintain the same
    language, structure and style of the
     article.

Article:
<replace article here>
```

B.3 XNLI data prompt

To include XNLI datapoints within the training dataset, we use the text in Listing 9 as the prompt and the text in Listing 10 as the accepted judgment.

Listing 9: Prompt for XNLI data

```
You will be given two sentences, a
    premise and a hypothesis. Your task
    is to determine whether the premise
    implies, contradicts, or neither
    implies nor contradicts the
    hypothesis.

Premise: <replace premise here>
Hypothesis: <replace hypthesis here>
```

Listing 10: Accepted judgment for XNLI data

```
<Based on XNLI datapoint label, choose
    one of:
"The premise implies the hypothesis"
"The premise contradicts the hypothesis"
"The premise neither implies nor
    contradicts the hypothesis">
```

C Additional framework description

We visually describe the direct and indirect strategies for generating summaries in Figure 4. We summarize the STEMF framework with the algorithm below.

Algorithm 1 STEMF: Self-Taught Evaluator for Multilingual Faithfulness

Input: Multilingual documents Ω, auxiliary LLM $\mathcal{M}$
Output: Multilingual self-taught evaluator $\mathcal{J}_r$
$\mathcal{J}_0 \leftarrow M$
for $i = 1$ to r **do**
  $\mathcal{A}_i \leftarrow \text{sample}(\Omega)$ ▷ Step 0
  $\mathcal{B}_i \leftarrow \emptyset, \mathcal{C}_i \leftarrow \emptyset$
  for each $d \in \mathcal{A}_i$ **do** ▷ Step 1
    $\{s_k\}_{k=1}^{K} \leftarrow M_{\text{Summarize}}(d)$
    for each sentence s_k **do**
      $\mathcal{B}_i \leftarrow \mathcal{B}_i \cup \{(d, s_k, 1)\}$
      $\widetilde{s}_k = M_{\text{Corrupt}}(s_k)$
      $\mathcal{B}_i \leftarrow \mathcal{B}_i \cup \{(d, \widetilde{s}_k, 0)\}$
    end for
  end for
  for each (d, s, l) in $\mathcal{B}_i$ **do** ▷ Step 2
    $(e, \widehat{l}) \leftarrow \mathcal{J}_{\text{judge}}(d, s)$
    if $\widehat{l} = l$ **then**
      $\mathcal{C}_i \leftarrow \mathcal{C}_i \cup (d, s, l)$
    end if
  end for
  $\mathcal{J}_i \leftarrow \text{Finetune}(\mathcal{J}_{i-1}, \mathcal{C}_i)$ ▷ Step 3
end for

D Evaluation metrics

We give here a reminder on the definition of balanced accuracy. Let $\mathcal{A}$ be a set of predictions, made of the set of true positives TP, the set of false positives FP, the set of true negatives TN, and the set of false negatives FN. Let the true positive rate be defined as $|TP|/(|TP| + |FN|)$ and the true

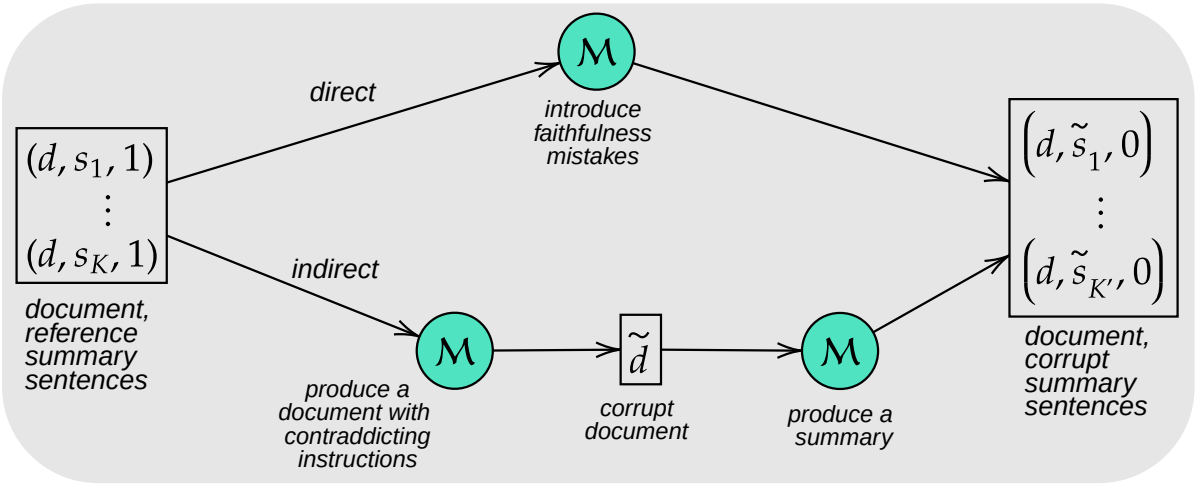

Figure 4: Scheme for producing corrupted sentences, given a document and faithful sentences. Replaces the "corrupt" arrow in Figure 1.

negative rate be defined as $|TN|/(|TN| + |FP|)$. The balanced accuracy is defined as the average between the true positive rate and true negative rate.

E Implementation details

We use `vllm` (Kwon et al., 2023) for inference and `alignment-handbook` (Tunstall et al.) for training the LLMs, adding some modifications to allow training only the central layers. Our experiments are run on 4 Nvidia H100 GPUs, when fine-tuning `gemma-2-9b`, or on 4 Nvidia L40S GPUs, in all other cases. We provide the hyperparamers used for training in Table 5 and the hyperparameters used for inference on `vllm` in Table 6.

Parameter	Value
Number of epochs	1
Gradient Accumulation Step	16
Batch Size	4
Learning rate	5e-5
LoRA Rank	128
LoRA Alpha	256
Lora Dropout	0.05
Max length	2048

Table 5: Hyper-parameter settings for LLM training.

F Tables

We provide here tables with detailed results on our experiments.

Parameter	Value
Temperature	1
Temperature (for judgments)	0
Max tokens	4096
Top_p	0.8

Table 6: Hyper-parameter settings for LLM inference.

	with XNLI	without XNLI
all	6.7	10.0
euro5	7.1	9.5
en	12.4	12.3
Average	8.8	10.6

Table 7: Improvement in average balanced accuracy, in percentage, over the starting model for `Qwen2.5-7B-Instruct` trained with or without XNLI data, on three combinations of training languages.

	1	3	5
Mistral-Nemo-Instruct-2407	7.5	16.1	14.1
Qwen2.5-3B-Instruct	3.7	4.5	4.5
Qwen2.5-7B-Instruct	7.8	12.1	12.3
gemma-2-2b-it	9.8	13.3	12.8
gemma-2-9b-it	2.4	2.3	2.9
Average	5.2	8.2	7.8

Table 8: Impact of the number of training iterations on model performance. The LLMs are trained using English (best language for training) and the performance is measured by absolute improvement in average balanced accuracy across languages and datasets.

	English	French	German	Hindi	Spanish	Arabic
	MEMERAG					mFACE
Models trained with our method STEMF						
Qwen-2.5-7B-Instruct	69.6	63.5	57.9	57.5	61.6	64.2
+ STEMF	76.4±0.9	71.7±0.6	68.1±1.0	75.9±2.4	75.2±1.5	73.5±1.1
gemma-2-9b-it	76.3	72.1	67.9	69.1	73.8	72.8
+ STEMF	74.4±1.8	**75.9±0.7**	66.3±1.9	**81.9±0.2**	77.6±1.2	70.9±0.8
SOTA						
MiniCheck-7B	79.1	66.1	71.5	73.0	73.2	73.4
Qwen2.5-72B-Instruct	**79.6**	75.6	**76.1**	76.7	**81.0**	**76.7**

	Italian	English	Avg
	FRANK		
Models trained with our method STEMF			
Qwen-2.5-7B-Instruct	68.4	75.0	65.1
+ STEMF	81.0±1.2	84.0±1.3	75.7±1.3
gemma-2-9b-it	81.5	85.5	75.6
+ STEMF	84.4±0.7	85.7±0.7	77.5±0.8
SOTA			
MiniCheck-7B	81.6	84.6	76.1
Qwen2.5-72B-Instruct	**85.6**	**87.5**	**80.0**

Table 9: Same as Table 1, but we report the average and standard error for three separate runs of STEMF. Balanced accuracy (in percentage) of our best trained evaluators, compared to the starting LLMs and strong baselines. The evaluations on English, French, German, Hindi, and Spanish are done on the MEMERAG dataset, while the evaluations for Arabic and Italian are done on the mFACE and FRANK datasets, respectively. We also report the performance on the original English FRANK. The Avg column reports the balanced accuracy averaged over all datasets and languages. We report in bold and underlined the highest and second highest values, respectively, for each column.

Name	Native languages	Translated to	# annotators	# factual	# non-factual
FRANK	en	ar, de, es, fr, it	3	3063	1829
MEMERAG	de, en, es, fr, hi	/	5	986	314
mFACE	ar, en, es, fr, hi	/	3	1362	1931

Table 10: Details on evaluation benchmarks.

	Arabic	German	English	Spanish	French	Hindi	Italian
Qwen2.5-3B-Instruct	44.0	53.7	65.6	56.9	56.2	39.1	54.7
Qwen2.5-7B-Instruct	56.2	62.1	74.2	65.6	65.0	48.5	65.3
Qwen2.5-72B-Instruct	74.3	72.5	86.1	77.5	76.0	69.1	72.5
gemma-2-2b-it	36.0	47.0	56.1	48.0	47.4	38.3	46.1
gemma-2-9b-it	57.7	64.5	72.3	66.9	66.6	57.6	66.6
Mistral-Nemo-Instruct-2407	47.8	53.9	68.0	58.9	59.5	47.1	58.9

Table 11: MMMLU scores for the models considered in this work.

	all	euro5	German	English	Spanish	French	Hindi
Mistral-Nemo-Instruct-2407	9.3	11.8	14.3	14.1	10.6	15.2	12.3
Qwen2.5-3B-Instruct	3.6	0.9	-10.0	4.5	1.6	-0.1	-4.2
Qwen2.5-7B-Instruct	10.0	9.5	9.0	12.3	8.3	8.5	4.4
gemma-2-2b-it	15.5	14.6	15.4	12.8	15.2	15.0	15.0
gemma-2-9b-it	1.3	2.1	2.1	2.9	1.9	1.7	1.0
Average	6.2	6.2	4.7	7.8	6.1	6.1	4.4

Table 12: Improvement in balanced accuracy, in percentage points, over the starting model, for the considered starting model and training languages combinations. The balanced accuracy is averaged over all benchmarks and languages. The models are trained using the base methodology and the indirect corruption strategy.

	all	euro5	German	English	Spanish	French	Hindi
Mistral-Nemo-Instruct-2407	68.0	70.5	72.9	72.7	69.2	73.9	70.9
Qwen2.5-3B-Instruct	72.2	69.5	58.6	73.1	70.3	68.5	64.4
Qwen2.5-7B-Instruct	75.1	74.6	74.2	77.4	73.4	73.6	69.5
gemma-2-2b-it	72.0	71.1	71.9	69.3	71.7	71.5	71.5
gemma-2-9b-it	76.9	77.7	77.7	78.5	77.5	77.3	76.6
Average	72.6	72.7	71.1	74.2	72.5	72.6	70.9

Table 13: Final balanced accuracy, in percentage, for the considered starting model and training languages combinations. The balanced accuracy is averaged over all benchmarks and languages. The models are trained using the base methodology and the indirect corruption strategy.