Towards Modular Fine-tuning of LLM-based Multilingual Neural Machine Translation

Anonymous ACL submission

Abstract

Multilingual Fine-tuning of Large Language Models (LLMs) has achieved great advancements in machine translation. However, existing research only focuses on a single-round fine-tuning process with fixed training data, often lacking adaptability, where introducing new languages will influence the performance of existing ones. In this study, we propose a modular fine-tuning pipeline that enables dynamic language support for LLMs. Instead of directly fine-tuning on all languages, our approach first trains English-centric LoRA adapters for each input and output language separately, and then merges the corresponding adapters' parameters without any extra training during inference. Experiments on 12 translation directions involving four low-resource and less-supported languages show that modular fine-tuning achieves up to 86% performance of traditional multi-parallel full-parameter fine-tuning, while training only 0.1% parameters and relying solely on English-centric data. Furthermore, we perform a comprehensive analysis about the merging ratio, when to merge, and the rationale for using English as a bridge language via Bayesian Optimization and logit lens.[1]

1 Introduction

Recent advances in Multilingual Neural Machine Translation (MNMT) (Xu et al., 2024a,b; Alves et al., 2024) have significantly reduced the performance gap between Large Language Models (LLMs) and conventional translation models (Team et al., 2022). However, existing research mainly focuses on the traditional fine-tuning settings with fixed training data, where all languages share the same trainable parameters. This often leads to negative language interference (Duh et al., 2012; Chen et al., 2023a; Huang et al., 2023) and also limits the model's scalability to new languages. The dynamic language adaptation for LLMs remains an underexplored problem.

[1]Codes are available at `https://anonymous.4open.science/r/MMT-4353/`.

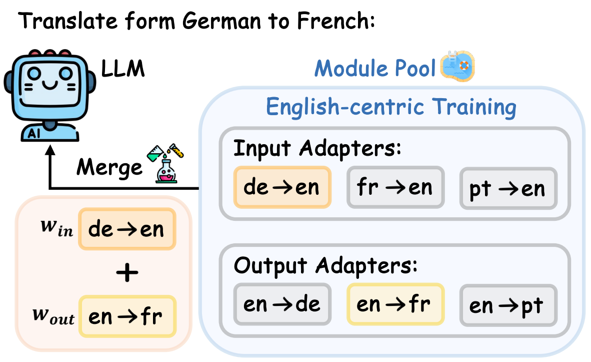


Figure 1: Example of our proposed modular fine-tuning pipeline on three languages: German (de), French (fr), Portuguese (pt). We first train English-centric adapters (x → eng or eng → x) for input and output languages separately. Then directly merge the parameters of corresponding input and out adapters from the module pool during a specific translation direction, e.g., translating from German to French.

To alleviate the interference and enable more flexible language adaptation, there is an increasing tendency in current research to leverage the modular nature of language models (Xiao et al., 2024) to isolate different languages into separate language-specific modules, e.g., extracting sub-networks (He et al., 2023; Tan et al., 2024) for different languages from the original model, or introducing language-specific adapters (Pfeiffer et al., 2022; Pires et al., 2023; Cao et al., 2024; Xu et al., 2025). Despite their effectiveness, these language-specific modules are still shared across different language directions and interfere with each other during cross-lingual training. We argue that these approaches do not achieve complete modularity, as the training of these modules is not fully independent. As a result, the model still cannot dynamically adapt to new languages, given that retraining is necessary for all existing language modules.

Therefore, in this work, we dive deep into MNMT under a *Completely Modular Fine-tuning* setting, where language-specific adapters share no parameters and do not influence each other during training. More specifically, we propose a *Compeletely Modular Fine-tuning* pipeline, which first trains English-centric adapters for input and output languages separately, and then directly merges the corresponding adapter parameters without any further training during translation. As shown in Figure 1, since all adapters are trained in a single direction, i.e., translating from or to English, each language adapter can be trained independently. This enables the dynamic language support, as introducing a new language only requires training the corresponding English-centric adapters without any influence on existing ones. For example, to enable German support, we only need to train the German-to-English input adapter and the English-to-German output adapter. By combining them with other existing adapters, the model can then translate in all directions involving German.

We conduct experiments on 12 language directions of 4 low-resource and less-supported languages from the FLORES+ (NLLB Team et al., 2024) dataset and compare the performance with traditional multi-parallel full-parameter fine-tuning settings. Results show that our method can achieve up to 86% performance of full-parameter fine-tuning on Llama3-8B-Instruct while only using 0.1% trainable parameters and also reducing the training time as we rely solely on English-centric data. Furthermore, we conduct a comprehensive analysis about the merging ratio, when to merge, and the rationale for using English as the bridge language via Bayesian optimization and logit lens.

2 Backgrounds

Multilingual Machine Translation Given a set of n languages $\mathbb{L} = \{l_1, l_2, \cdots, l_n\}$, multilingual machine translation aims to translate an input sentence $\boldsymbol{x}$ in the source language $in \in \mathbb{L}$ into an output sentence $\boldsymbol{y}$ in the target language $out \in \mathbb{L}$. With an MNMT dataset including N sentence pairs $\mathbb{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) : i \in 1 \cdots N\}$, the training loss is defined as:

$$\mathcal{L}_{MNMT} = - \sum_{\boldsymbol{x},\boldsymbol{y} \in \mathbb{D}} \sum_{j=1}^{J} \log\ p_\theta(y_j | \boldsymbol{y}_{<j}, \boldsymbol{x}) \quad (1)$$

where $\boldsymbol{x} = x_1, x_2, \cdots, x_I$ is a input sentence with length I and $\boldsymbol{y} = y_1, y_2, \cdots, y_J$ is the corresponding output sentence with length J.

Low Rank Adaptation (LoRA) LoRA (Hu et al., 2022) is widely used in Parameter-efficient Fine-tuning (PEFT) for Large Language Models where fine-tuning is re-parameterized in a low-rank intrinsic subspace. Given a weight matrix in a pre-trained model $\mathbf{W} \in \mathbb{R}^{d \times k}$, LoRA forward pass can be calculated as:

$$\boldsymbol{h} = \mathbf{W}\boldsymbol{x} + \mathbf{B}\mathbf{A}\boldsymbol{x} \quad (2)$$

where $\mathbf{B} \in \mathbb{R}^{d \times r}$ and $\mathbf{A} \in \mathbb{R}^{r \times k}$. During training, $\mathbf{W}$ will be frozen and the trainable parameters, i.e., $\mathbf{A}$ and $\mathbf{B}$, will be reduced from $d \times k$ to $d \times r + r \times k$, where $r \ll min(d, k)$. We choose LoRA as the architecture of language adapters in this research, thanks to its parameter efficiency and flexibility.

3 MNMT from a Modular Perspective

3.1 Completely Modular Fine-tuning

We assess the degree of modularity in an MNMT system based on two factors: whether different languages share any trainable parameters, and whether language-specific adapters influence each other during training. If there is no parameter sharing across languages and the training processes are entirely independent, we refer to the setup as *Completely Modular Fine-tuning*.

We want to emphasize that although some previous work (Pires et al., 2023; Cao et al., 2024) also avoids parameter sharing across languages by separating them into input and output adapters, these adapters still influence each other during multilingual training. Consider a translation from an input language l_{in} to an output language l_{out}, the corresponding adapters for l_{in} and l_{out} are loaded simultaneously during training. As a result, the training of these adapters is still not independent, hindering the scalability of the model to new languages.

To enable *Completely Modular Fine-tuning*, we decompose the MNMT task into input and output language modules, and propose a modular fine-tuning pipeline. As illustrated in Figure 1, we first train English-centric modules and then directly merge the corresponding ones to translate between any languages.

3.2 English-centric Module Training

We use LoRA as the architecture of our language module adapter. For each language $l_i \in \mathbb{L}$, we build an input adapter $\text{LoRA}_{in}^{l_i}$ and an output adapter $\text{LoRA}_{out}^{l_i}$, resulting in a module pool with $2n$ adapters. As shown in Figure 1, all adapters are trained separately in an English-centric setting, where $\text{LoRA}_{in}^{l_i}$ is only trained on the direction from l_i to English, and $\text{LoRA}_{out}^{l_i}$ is only trained on the direction from English to l_i.

We choose English as the bridge language based on the intuition that English often serves as the latent language in current LLMs (Wendler et al., 2024; Kargaran et al., 2025). We expect that using English as the bridge language can maximize the cross-lingual transfer during training. Another advantage of this design is the improved data efficiency, since it requires only English-centric translation data. Non-English directions often lack sufficient parallel data, especially for those low-resource languages (Goyal et al., 2022).

3.3 Merging

As mentioned in Section 2, we use LoRA as the adapter architecture. Each language module is composed of two matrices, denoted as $\mathbf{A}$ and $\mathbf{B}$. When translating from an input language l_{in} to an output language l_{out}, only the corresponding input and output adapters $\text{LoRA}_{in}^{l_{in}}$ and $\text{LoRA}_{out}^{l_{out}}$ are activated. During translation, we directly merge the parameters of $\mathbf{A}$ and $\mathbf{B}$ separately with Equation 3 without any training. We adopt a weighted merging strategy, following the implementation in the PEFT library,[2] where $w_{\text{in}} + w_{\text{out}} = 1$ and are shared across all language directions.

$$\begin{aligned}\mathbf{B}_{merge} &= (\sqrt{w_{in}}\mathbf{B}_{in}^{l_{in}} + \sqrt{w_{out}}\mathbf{B}_{out}^{l_{out}}) \\ \mathbf{A}_{merge} &= (\sqrt{w_{in}}\mathbf{A}_{in}^{l_{in}} + \sqrt{w_{out}}\mathbf{A}_{out}^{l_{out}})\end{aligned} \quad (3)$$

where $\mathbf{B}_{in}^{l_{in}}$, $\mathbf{B}_{out}^{l_{out}}$, $\mathbf{A}_{in}^{l_{in}}$, $\mathbf{A}_{out}^{l_{out}}$ are the low-rank matrices of the activated $\text{LoRA}_{in}^{l_{in}}$ and $\text{LoRA}_{out}^{l_{out}}$. Then, the forward pass during inference can be calculated as:

$$y = \mathbf{W}x + \mathbf{B}_{merge}\mathbf{A}_{merge}x, \quad (4)$$

where $\mathbf{W}$ is a given weight matrix in the original model.

Our approach satisfies the requirements of *Completely Modular Fine-tuning*, given that all adapters are trained independently and merged without any additional training.

[2]https://huggingface.co/docs/peft/index

4 Experimental Setup

Dataset FLORES+ (NLLB Team et al., 2024) is a high-quality multi-parallel dataset supporting translation between over 200 languages. All sentences are divided into three splits: dev (997 sentences), devtest (1,012 sentences), and test (992 sentences). Since the test set is not publicly available, we use the dev set for training and devtest set for evaluation.

Backbone Model and Metric We first adopt Qwen2.5-0.5B-Instruct (Qwen et al., 2025) as the backbone model, thanks to Qwen's strong multilingual support. We further extend our experiments to larger models, including Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct (Grattafiori et al., 2024), to verify the scalability of our method. We choose chrF++ (Popović, 2017) as our evaluation metric in the following experiments. Although neural-based evaluation metrics, such as COMET (Rei et al., 2020, 2022; Guerreiro et al., 2024) are widely used given their higher agreement with human judgments, this research mainly focuses on low-resource languages which LLMs perform poor, and such metrics are often unreliable in low-resource settings. Similarly, BLEU score (Papineni et al., 2002) is also sensitive to tokenization (Post, 2018; Goyal et al., 2022), leading to inconsistent evaluation for low-resource languages that lack standardized tokenizers. Therefore, we use chrF++, as it performs well on typologically diverse languages. We use greedy decoding during evaluation for higher efficiency.

Language Selection While prior work often groups languages by resource level, we argue that the resource level cannot accurately reflect an LLM's ability for a language. On the one hand, an LLM's performance on a given language often correlates with its training data, and some low-resource languages may perform well due to targeted training. On the other hand, some low-resource languages, e.g., Occitan, may also achieve strong performance due to their high similarity with certain high-resource languages (Team et al., 2022; Cao et al., 2024). To fairly compare the performance of modular fine-tuning on well-supported and less-supported languages, we begin by evaluating the English-centric performance of Qwen2.5-0.5B-Instruct on 179 languages in the FLORES+ dataset, excluding those without a dev or devtest set. As shown in Figure 2, we report

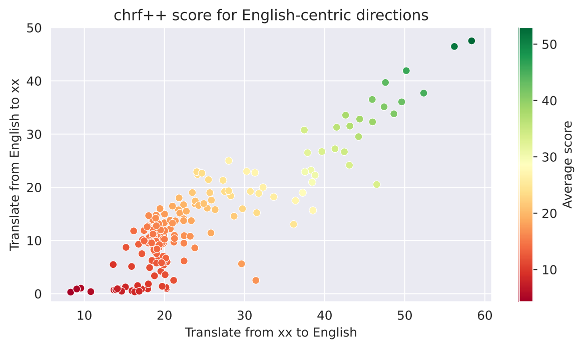


Figure 2: English-centric chrF++ scores of Qwen2.5-0.5B-Instruct on 179 languages in the FLORES+ dataset. Each point represents a language, with the x-axis showing performance for translation from the language into English, and the y-axis showing translation from English into the language. The color indicates the average of the two directions, which we use to estimate the model's ability on each language.

the English-centric chrF++ scores on the Qwen2.5-0.5B-Instruct model and divide all languages into three groups based on their average scores s translating from and to English: high-performance ($\geq$ 40), medium-performance (20 $< s <$ 40), and low-performance ($\leq$ 40). We randomly choose four languages from each group for experiments. More details about the selected languages are provided in Appendix A.

Baseline For the four languages in each group, we evaluate translation performance on all 12 directions. We choose traditional full-parameter fine-tuning with multi-parallel data of all 12 directions as a strong baseline.

Training Details All experiments of Qwen2.5-0.5B-Instruct are conducted on a single NVIDIA A100 40GB GPU. For Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct, we train LoRA modules on a single NVIDIA H200 GPU. Full-parameter fine-tuning is performed using DeepSpeed ZeRO-3 (Rajbhandari et al., 2020), and all experiments use bfloat16 (bf16) precision. For full-parameter fine-tuning, we report the best score across all epochs as our baseline. For LoRA modules, we choose the best checkpoint based on the English-centric performance for merging. We apply LoRA to the weight matrices of both the attention (q, k, v) and the MLP (fc1, fc2) across all layers. We use supervised fine-tuning (SFT) and calculate the loss of output sentences during training. More details are provided in Appendix B.

5 Main Results

Table 1 shows the chrF++ scores of four low-performance languages on Qwen2.5-0.5B-Instruct: Moroccan Tamazight (zgh), Tamasheq (taq), Acehnese (ace), Minangkabau (min). We evaluate all 12 translation directions and report the average score of the three directions when translating to ($\rightarrow l$) or from ($l \rightarrow$) a specific language l.

We first conduct experiments on Qwen2.5-0.5B-Instruct using different LoRA ranks r and merging ratios, denoted as $w_{in} : w_{out}$. We use green to highlight scores that exceed 75% of traditional full-parameter fine-tuning performance, and underline scores where merging outperforms using only LoRA$_{in}$ or LoRA$_{out}$. As shown in Table 1, performance consistently improves with increasing rank. Merge (1:9,r=64) achieves 79% of the full-parameter baseline performance while using only 8% of the trainable parameters. Notably, compared to the traditional multi-parallel baseline, modular fine-tuning is almost zero-shot, given these adapters are trained using only a single direction, translating from or to English, and all 12 evaluated directions are never seen during training.

Although only using LoRA$_{in}$ performs poorly due to being trained exclusively on to-English direction, we surprisingly find that a simple merge of LoRA$_{in}$ and LoRA$_{out}$ yields consistent improvements in almost all directions. Furthermore, the results also indicate that the output language plays a much more important role than the input language in machine translation. A merging ratio of $w_{in} : w_{out} = 1 : 9$ is the only setting that consistently brings improvements across all ranks, while further increasing the weight of the input language hurts performance. We believe this is because, for language models, language understanding is much easier than language generation (Li et al., 2024). Therefore, the model does not require much input language information to perform well.

We further extend our experiments to two larger models: Qwen2.5-7B-Instruct (Table 2) and LLaMA3.1-8B-Instruct (Table 3). With just 0.1% trainable parameters, our method reaches 86% of the full-parameter performance on Llama3.1-8B-Instruct and 68% on Qwen2.5-7B-Instruct. Consistent with previous experiments, merging LoRA$_{in}$ and LoRA$_{out}$ brings improvements on almost all directions. Additionally, our method can be naturally extended to new languages without influencing existing ones, and it requires only English-centric

Methods	Language Directions								AVG
	ace→	min→	taq→	zgh→	→ace	→min	→taq	→zgh	
Pre-train	0.94	1.00	0.75	0.89	0.95	0.51	1.06	1.07	0.90
Full-parameter	**17.73**	**16.85**	**14.83**	**15.24**	**14.22**	**16.47**	**16.82**	**17.14**	**16.16**
Only LoRA$_{in}$(r=8)	1.17	1.40	0.92	1.05	1.00	0.80	1.46	1.27	1.13
Only LoRA$_{out}$(r=8)	9.99	7.56	6.48	6.46	8.71	9.91	4.14	7.71	7.62
Merge(1:9,r=8)	12.22	12.71	8.19	8.49	11.76	12.24	8.21	9.39	10.40
Merge(3:7,r=8)	11.10	11.57	8.18	7.03	12.47	11.96	5.08	8.38	9.47
Merge(5:5,r=8)	8.08	6.79	5.14	2.43	6.97	9.71	1.73	4.03	5.61
Merge(7:3,r=8)	3.83	1.72	1.47	1.19	1.74	4.20	1.30	0.97	2.05
Merge(9:1,r=8)	1.06	1.34	0.93	1.08	0.99	0.82	1.46	1.13	1.10
Only LoRA$_{in}$(r=16)	1.21	1.48	0.96	1.11	1.05	0.82	1.52	1.37	1.19
Only LoRA$_{out}$(r=16)	10.11	6.32	5.56	7.91	9.31	10.35	7.24	3.01	7.47
Merge(1:9,r=16)	12.19	12.19	9.30	8.43	10.06	11.52	9.39	11.14	10.53
Merge(3:7,r=16)	10.98	11.17	7.17	6.59	8.79	10.91	6.52	9.69	8.98
Merge(5:5,r=16)	7.74	5.96	4.52	4.69	5.45	9.14	2.18	6.14	5.73
Merge(7:3,r=16)	3.03	1.34	1.76	1.11	0.97	3.36	1.43	1.48	1.81
Merge(9:1,r=16)	1.24	1.58	1.02	1.13	1.09	0.85	1.54	1.49	1.24
Only LoRA$_{in}$(r=32)	1.33	1.49	0.99	1.17	1.07	0.83	1.58	1.50	1.25
Only LoRA$_{out}$(r=32)	13.83	9.05	8.70	9.35	10.00	11.18	9.77	9.97	10.23
Merge(1:9,r=32)	13.29	13.23	10.30	10.28	10.99	13.61	10.26	12.23	11.77
Merge(3:7,r=32)	12.02	12.80	8.83	7.99	11.16	13.35	6.78	10.35	10.41
Merge(5:5,r=32)	7.91	7.11	6.72	6.27	7.99	11.65	1.66	6.71	7.00
Merge(7:3,r=32)	2.95	1.59	4.33	1.38	1.83	5.27	1.59	1.57	2.56
Merge(9:1,r=32)	1.35	1.55	1.11	1.23	1.17	0.84	1.62	1.61	1.31
Only LoRA$_{in}$(r=64)	1.34	1.51	1.04	1.14	1.12	0.83	1.56	1.53	1.26
Only LoRA$_{out}$(r=64)	14.41	12.04	9.42	10.13	10.41	12.09	10.64	12.86	11.50
Merge(1:9,r=64)	14.40	14.57	11.11	11.80	10.47	14.37	12.63	14.40	12.97
Merge(3:7,r=64)	12.76	13.91	8.79	9.49	11.12	13.47	9.47	10.88	11.24
Merge(5:5,r=64)	8.75	7.67	6.74	7.99	9.10	10.90	4.98	6.19	7.79
Merge(7:3,r=64)	2.79	2.30	5.41	5.76	5.91	7.07	1.56	1.71	4.06
Merge(9:1,r=64)	1.43	1.63	1.20	1.22	1.25	0.98	1.61	1.63	1.37

Table 1: The chrF++ scores for four low-performance languages on Qwen2.5-0.5B-Instruct. We evaluate all 12 directions and report the average score translating to ($\rightarrow l$) or from ($l \rightarrow$) a specific language l. We also explore different merging ratios and LoRA ranks r, denoted as Merge(w_{in}:w_{out}, r). Scores where merging outperforms using only LoRA$_{in}$ or LoRA$_{out}$ are underlined, and those achieving more than 75% of the full-parameter fine-tuning performance are highlighted in green.

data, demonstrating the potential of modular fine-tuning. We also notice that Llama3.1 performs better than Qwen2.5. We attribute this to Llama's stronger English-centric nature compared to Qwen, which allows it to benefit more when we use English as the bridge language. We provide further analysis in Section 6.3.

6 Analysis and Discussion

6.1 Is There an Optimal Merging Ratio?

In previous experiments, we tested several merging ratios intuitively. This naturally leads to the following question: Is there an optimal merging weight setting? To answer this question, we perform a case study on translating from Acehnese (ace) to Minangkabau (min), where we apply Bayesian Optimization (Gardner et al., 2014; Nogueira, 2014) to search for the optimal merging weights.

Given a layer index i, we can denote the weight of LoRA$_{out}$ at that layer as w^i_{out}. The weight of LoRA$_{in}$ at that layer can be calculated as:

$$w^i_{in} = 1 - w^i_{out}. \tag{5}$$

To reduce the hyperparameter search space, we assume that the w_{out} for each layer follows a linear relationship with respect to the layer index i:

$$w^i_{out} = ai + b, \tag{6}$$

given specific values of a and b, all w^i_{in} and w^i_{out} can be computed directly. During Bayesian Optimization, we set the LoRA$_{out}$ weights for the first layer w^{first}_{out} and the last layer w^{last}_{out} as the only two

Methods	Language Directions								AVG
	ace→	min→	taq→	zgh→	→ace	→min	→taq	→zgh	
Pre-train	2.76	2.02	1.18	1.03	1.92	2.43	1.21	1.43	1.75
Full-parameter	**19.44**	**18.87**	**17.11**	**17.33**	**17.31**	**19.40**	**17.95**	**18.09**	**18.19**
Only LoRA$_{in}$(r=8)	1.60	1.96	1.11	1.18	1.57	1.11	1.73	1.45	1.46
Only LoRA$_{out}$(r=8)	13.94	11.88	11.06	11.83	12.45	13.91	12.59	9.75	12.18
Merge(1:9,r=8)	$\underline{14.48}$	$\underline{14.46}$	$\underline{11.30}$	$\underline{12.06}$	$\underline{13.05}$	$\underline{14.59}$	11.88	$\underline{12.76}$	$\underline{13.07}$
Merge(3:7,r=8)	12.82	$\underline{12.56}$	10.80	11.07	$\underline{13.05}$	$\underline{14.51}$	8.97	$\underline{10.71}$	11.81
Merge(5:5,r=8)	9.29	7.76	9.56	9.18	11.54	13.24	4.37	6.65	8.95
Merge(7:3,r=8)	4.88	4.68	4.84	4.68	6.90	9.09	1.46	1.62	4.77
Merge(9:1,r=8)	2.42	2.16	1.15	3.09	3.17	2.87	1.58	1.21	2.21

Table 2: The chrF++ scores for four low-performance languages on Qwen2.5-7B-Instruct. The merging ratio and the LoRA rank are denoted as w_{in}:w_{out} and r respectively. Scores where merging outperforms using only LoRA$_{in}$ and LoRA$_{out}$ are underlined.

Methods	Language Directions								AVG
	ace→	min→	taq→	zgh→	→ace	→min	→taq	→zgh	
Pre-train	1.54	1.48	1.35	1.44	1.26	1.19	1.66	1.69	1.45
Full-parameter	**19.70**	**19.15**	**17.00**	**17.35**	**17.64**	**19.45**	**18.20**	**17.91**	**18.30**
Only LoRA$_{in}$(r=8)	1.28	1.36	1.17	1.27	1.24	0.90	1.52	1.41	1.27
Only LoRA$_{out}$(r=8)	17.70	16.62	11.59	11.99	11.81	13.10	16.86	16.13	14.47
Merge(1:9,r=8)	$\underline{18.26}$	$\underline{17.43}$	$\underline{12.79}$	$\underline{13.67}$	$\underline{13.41}$	$\underline{14.98}$	16.86	$\underline{16.90}$	$\underline{15.54}$
Merge(3:7,r=8)	$\underline{18.16}$	$\underline{17.41}$	$\underline{13.73}$	$\underline{14.72}$	$\underline{15.68}$	$\underline{16.62}$	15.33	$\underline{16.38}$	$\underline{16.00}$
Merge(5:5,r=8)	17.31	15.96	$\underline{12.70}$	$\underline{12.61}$	$\underline{15.58}$	$\underline{16.95}$	11.79	14.27	$\underline{14.65}$
Merge(7:3,r=8)	15.62	12.98	8.03	9.34	$\underline{12.52}$	$\underline{15.34}$	8.07	10.03	11.49
Merge(9:1,r=8)	5.68	3.09	1.19	1.31	1.43	1.47	2.72	5.63	2.81

Table 3: The chrF++ scores for four low-performance languages on Llama3.1-8B-Instruct. The merging ratio and the LoRA rank are denoted as w_{in}:w_{out} and r respectively. Scores where merging outperforms using only LoRA$_{in}$ and LoRA$_{out}$ are underlined.

hyperparameters to be searched. Given w_{out}^{first} and w_{out}^{last}, the linear coefficients a and b can be computed as:

$$b = w_{out}^{first} \tag{7}$$

$$a = \frac{w_{out}^{last} - w_{out}^{first}}{last}. \tag{8}$$

We choose the setting of Qwen2.5-0.5B-Instruct, Merge(1:9, r=8) as our baseline, where the score for translating from Acehnese to Minangkabau (ace→min) is 17.05. After 50 iterations of Bayesian Optimization, we obtain a higher score of 17.54, with $w_{out}^{first} \approx 0.99$, $w_{out}^{last} \approx 0.91$. We then extend this setting to all other directions. However, as shown in Table 4, this setting (Bayesian (r=8)) fails to achieve a higher overall score. Considering the inherently multi-objective nature of MNMT, i.e., improving performance for some languages may come at the expense of others (Duh et al., 2012; Huang et al., 2023; Chen et al., 2023b), it's almost impossible to find an optimal setting that works well for all languages. As shown in the results, while hyperparameter search can improve the performance for a specific language, it inevitably degrades performance for others. The 1:9 ratio provides a relatively good trade-off. This is in line with our earlier conclusion that LoRA$_{out}$ plays a much more important role during translation.

6.2 Results on High- and Medium-performance languages

Previous experiments focused on four low-performance languages. In this section, we shift our attention to another four high-performance (French, Portuguese, Spanish, German) and four medium-performance languages (Czech, Asturian, Japanese, Ukrainian).

As shown in Appendix C, compared to low-performance languages, medium-performance languages (Table 7) can achieve comparable results to the full-parameter fine-tuning baseline using only

Methods	Language Directions								AVG
	ace→	min→	taq→	zgh→	→ace	→min	→taq	→zgh	
Pre-train	0.94	1.00	0.75	0.89	0.95	0.51	1.06	1.07	0.90
Full-parameter	**17.73**	**16.85**	**14.83**	**15.24**	**14.22**	**16.47**	**16.82**	**17.14**	**16.16**
Only InputLoRA(r=8)	1.17	1.40	0.92	1.05	1.00	0.80	1.46	1.27	1.13
Only OutputLoRA(r=8)	9.99	7.56	6.48	6.46	8.71	9.91	4.14	7.71	7.62
Merge(1:9,r=8)	12.22	12.71	8.19	8.49	11.76	12.24	8.21	9.39	10.40
Bayesian(r=8)	12.41	12.41	8.78	7.95	10.92	12.07	8.06	10.50	10.39

Table 4: The chrF++ scores on Qwen2.5-0.5B-Instruct using the merging ratio after Bayesian Optimization (Bayesian(r=8)). We compared it with the previous best ratio (Merge(1:9, r=8)). Scores where merging outperforms using only LoRA$_{in}$ and LoRA$_{out}$ are underlined.

the LoRA$_{out}$ module with a smaller rank (Only LoRA$_{out}$(r=4)). In the case of high-performance languages (Table 6), results are even better than the full-parameter fine-tuning baseline. Merging with LoRA$_{in}$ provides almost no additional gain. We attribute this to the model having already acquired sufficient knowledge for these strong-performance languages, making the LoRA$_{in}$ unnecessary. Therefore, for these languages, using only the LoRA$_{out}$ is enough for effective modular fine-tuning.

6.3 Uncovering the Latent Language in Machine Translation

In previous experiments, we intuitively adopted English as the bridge language for English-centric modular training. In this section, we take a closer look at the rationale for this choice. Inspired by Wendler et al. (2024); Zhong et al. (2025), we conduct further analysis by visualizing the latent language during translation. We focus on the following two questions: *(1) What is the latent language used by current LLMs during translation? (2) Will large-scale multilingual fine-tuning change the latent language of LLMs?*

Specifically, we follow the experimental setup from Wendler et al. (2024), analyzing the models' latent language via Single Word Translation. We use a 4-shot prompt template:

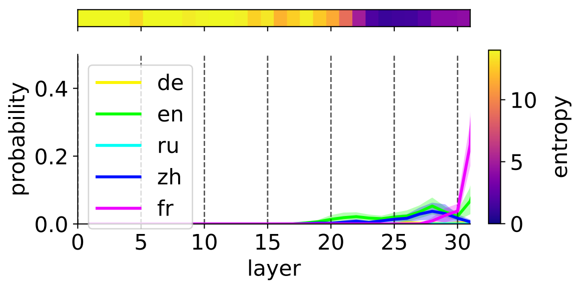
Template for Single Word Translation

Deutsch: "mutter" - Français: "mère"
Deutsch: "ozean" - Français: "océan"
Deutsch: "herz" - Français: "cœur"
Deutsch: "wort" - Français: "mot"
Deutsch: "berg" - Français: "

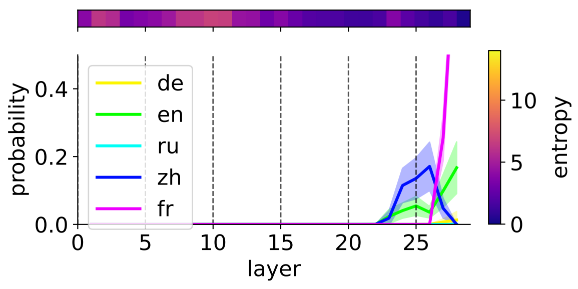
We do experiments on five languages: German (de), English (en), Russian (ru), Chinese (zh), and French (fr). Given the 4-shot prompt, the model will translate the fifth German word. We then apply the logit lens (Nostalgebraist, 2020) by feeding the hidden states from all layers into the language modeling head, and extract the probabilities for all the tokens corresponding to the given word in these five languages. This allows us to obtain the language probability at each layer.

To answer question (1), we conduct this analysis on Llama3.1-8B-Instruct and Qwen2.5-7B-Instruct. Figure 3 shows the language probability and entropy at each layer when translating from German (de) to French (fr). We find that the input language rarely serves as the latent language. Instead, the given word is first transformed into English and Chinese in the middle layers, and then translated into the output languages in the final layers. Both Llama and Qwen rely on English and Chinese as latent languages. Llama uses them in nearly equal proportions, with a slight preference for English, while Qwen shows a clear dominance of Chinese. This difference helps explain why Qwen performs worse than Llama in previous English-centric modular fine-tuning. It also suggests that we should consider the model's latent language preferences when choosing the bridge language. We report the analysis results for other directions in Appendix D.

For the second question, we compare the latent language between Llama2-13B (Touvron et al., 2023) and Alma-13B (Xu et al., 2024a). Alma is fine-tuned from Llama2 with billions of multilingual tokens. We choose Alma because it is one of the few LLMs that has undergone large-scale multilingual fine-tuning from a backbone LLM. This comparison allows us to investigate the impact of multilingual fine-tuning on the LLM's latent language. The results in Appendix D show that even after multilingual fine-tuning with billions



(a) Llama3.1-8B-Instruct



(b) Qwen2.5-7B-Instruct

Figure 3: The language probabilities when translating from German to French using Llama3.1-8B-Instruct and Qwen2.5-7B-Instruct. We show the probabilities on five languages at each layer: German (de), English (en), Russian (ru), Chinese (zh), French (fr). We also report the entropy on the top of the probabilities.

of tokens, the model's latent language tendencies remain largely unchanged. This indicates the traditional multilingual fine-tuning also aligns with the model's latent language preference, which is consistent with the idea of modular fine-tuning.

7 Related Work

LLM-based Machine Translation Current LLM-based machine translation research mainly focuses on improving performance by optimizing the training data and pipeline (Xu et al., 2024a; Alves et al., 2024), or by introducing reinforcement learning techniques (Xu et al., 2024b; Wu et al., 2024) to narrow the gap between LLMs and conventional translation-dedicated models (Fan et al., 2020; NLLB Team et al., 2024). However, they still focus on the traditional full-parameter fine-tuning setting with fixed training data, which introduces language interference and makes it difficult to extend to new languages. The dynamic language adaptation of LLMs remains underexplored.

Language-specific Learning There is also a series of studies that try to leverage the modular nature of language models for machine translation. Introducing language-specific structures is a common strategy in this kind of research. Sachan and Neubig (2018); Escolano et al. (2021); Pires et al. (2023) built language-specific encoder and decoder layers, and Cao et al. (2024) used LoRA as language-specific adapters to further reduce the trainable parameters. Another line of work (Lin et al., 2021; Wang and Zhang, 2022; He et al., 2023; Tan et al., 2024) tried to extract language-specific sub-networks from the language model. However, these studies only focus on alleviating language interference. The language-specific structures still influence each other during multilingual training, limiting the support of dynamic language adaptation.

Model Merging Model merging has recently emerged as a significant trend in the research of LLMs. Liu et al. (2025) merged intermediate checkpoints during pre-training to improve model performance, whereas our work targets the fine-tuning stage. Another line of work (Yu et al., 2024; Wan et al., 2024; Gupta and Gupta, 2024; Bandarkar et al., 2025) more closely related to ours aims to merge multiple expert models to enhance performance on a specific task. While they mainly focused on mathematics and coding, conducting model merging in multilingual settings receives little attention.

8 Conclusion

In this research, we studied the *Completely Modular Fine-tuning* setting for Multilingual Neural Machine Translation. We propose a modular fine-tuning pipeline that first trains English-centric LoRA adapters for each input and output language independently. Instead of traditional end-to-end training, we then merge the corresponding LoRA$_{in}$ and LoRA$_{out}$ to achieve the translation between any languages. We conduct experiments on 12 languages with different performance levels. We try varying merging ratios and find that LoRA$_{out}$ plays a much more important role than LoRA$_{in}$ during translation. For those languages which LLM performs poorly, merging can improve the performance across all directions, highlighting the potential of modular fine-tuning. We also notice that Llama performs better than Qwen in modular fine-tuning and attribute this to the stronger English-centric nature of Llama after analyzing the latent language. We hope our findings will encourage further research on modular training of LLMs.

Limitation

Despite the insights gained from our work, our research still has some limitations.

Firstly, we adopt a simple weighted-sum merging strategy for combining language-specific modules. While this approach is straightforward and effective, it may not fully exploit the potential of modular training. Exploring more sophisticated merging methods, or designing techniques that extract language-specific structures to reduce parameter overlap across modules, warrants further investigation.

Secondly, our experiments are conducted on relatively small-scale training data. There is always a trade-off between the amount of available data and language diversity. We believe that relying not only on parallel data but also incorporating other forms of data, such as monolingual corpora, may yield more insightful findings in future studies.

References

Duarte Miguel Alves, José Pombal, Nuno M Guerreiro, Pedro Henrique Martins, João Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, Sweta Agrawal, Pierre Colombo, José G. C. de Souza, and Andre Martins. 2024. Tower: An open multilingual large language model for translation-related tasks. In *First Conference on Language Modeling*.

Lucas Bandarkar, Benjamin Muller, Pritish Yuvraj, Rui Hou, Nayan Singhal, Hongjiang Lv, and Bing Liu. 2025. Layer swapping for zero-shot cross-lingual transfer in large language models. *Preprint*, arXiv:2410.01335.

Zhe Cao, Zhi Qu, Hidetaka Kamigaito, and Taro Watanabe. 2024. Exploring intrinsic language-specific subspaces in fine-tuning multilingual neural machine translation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21142–21157, Miami, Florida, USA. Association for Computational Linguistics.

Liang Chen, Shuming Ma, Dongdong Zhang, Furu Wei, and Baobao Chang. 2023a. On the pareto front of multilingual neural machine translation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Liang Chen, Shuming Ma, Dongdong Zhang, Furu Wei, and Baobao Chang. 2023b. On the pareto front of multilingual neural machine translation. *Preprint*, arXiv:2304.03216.

Kevin Duh, Katsuhito Sudoh, Xianchao Wu, Hajime Tsukada, and Masaaki Nagata. 2012. Learning to translate with multiple objectives. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1–10, Jeju Island, Korea. Association for Computational Linguistics.

Carlos Escolano, Marta R. Costa-jussà, José A. R. Fonollosa, and Mikel Artetxe. 2021. Multilingual machine translation: Closing the gap between shared and language-specific encoder-decoders. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 944–948, Online. Association for Computational Linguistics.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2020. Beyond english-centric multilingual machine translation. *Preprint*, arXiv:2010.11125.

Jacob R Gardner, Matt J Kusner, Zhixiang Eddie Xu, Kilian Q Weinberger, and John P Cunningham. 2014. Bayesian optimization with inequality constraints. In *ICML*, volume 2014, pages 937–945.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Nuno M. Guerreiro, Ricardo Rei, Daan van Stigt, Luisa Coheur, Pierre Colombo, and André F. T. Martins. 2024. xcomet: Transparent machine translation evaluation through fine-grained error detection. *Transactions of the Association for Computational Linguistics*, 12:979–995.

Siddharth Gupta and Aakash Gupta. 2024. Model merging using geometric median of task vectors. In *LLM Merging Competition at NeurIPS 2024*.

Dan He, Minh-Quang Pham, Thanh-Le Ha, and Marco Turchi. 2023. Gradient-based gradual pruning for language-specific multilingual neural machine translation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 654–670, Singapore. Association for Computational Linguistics.

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*.

Yichong Huang, Xiaocheng Feng, Xinwei Geng, Baohang Li, and Bing Qin. 2023. Towards higher Pareto frontier in multilingual machine translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3802–3818, Toronto, Canada. Association for Computational Linguistics.

Amir Hossein Kargaran, Ali Modarressi, Nafiseh Nikeghbal, Jana Diesner, François Yvon, and Hinrich Schütze. 2025. Mexa: Multilingual evaluation of english-centric llms via cross-lingual alignment. *Preprint*, arXiv:2410.05873.

Chong Li, Wen Yang, Jiajun Zhang, Jinliang Lu, Shaonan Wang, and Chengqing Zong. 2024. X-instruction: Aligning language model in low-resource languages with self-curated cross-lingual instructions. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 546–566, Bangkok, Thailand. Association for Computational Linguistics.

Zehui Lin, Liwei Wu, Mingxuan Wang, and Lei Li. 2021. Learning language specific sub-network for multilingual machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 293–305, Online. Association for Computational Linguistics.

Deyuan Liu, Zecheng Wang, Bingning Wang, Weipeng Chen, Chunshan Li, Zhiying Tu, Dianhui Chu, Bo Li, and Dianbo Sui. 2025. Checkpoint merging via bayesian optimization in llm pretraining. *Preprint*, arXiv:2403.19390.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, and 20 others. 2024. Scaling neural machine translation to 200 languages. *Nature*, 630(8018):841–846.

Fernando Nogueira. 2014. Bayesian Optimization: Open source constrained global optimization tool for Python.

Nostalgebraist. 2020. Interpreting gpt: The logit lens. LessWrong.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Jonas Pfeiffer, Naman Goyal, Xi Lin, Xian Li, James Cross, Sebastian Riedel, and Mikel Artetxe. 2022. Lifting the curse of multilinguality by pre-training modular transformers. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3479–3495, Seattle, United States. Association for Computational Linguistics.

Telmo Pires, Robin Schmidt, Yi-Hsiu Liao, and Stephan Peitz. 2023. Learning language-specific layers for multilingual machine translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14767–14783, Toronto, Canada. Association for Computational Linguistics.

Maja Popović. 2017. chrf++: words helping character n-grams. In *Proceedings of the second conference on machine translation*, pages 612–618.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

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, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. *Preprint*, arXiv:1910.02054.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Ricardo Rei, Marcos Treviso, Nuno M. Guerreiro, Chrysoula Zerva, Ana C Farinha, Christine Maroti, José G. C. de Souza, Taisiya Glushkova, Duarte Alves, Luisa Coheur, Alon Lavie, and André F. T. Martins. 2022. CometKiwi: IST-unbabel 2022 submission for the quality estimation shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 634–645, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Devendra Sachan and Graham Neubig. 2018. Parameter sharing methods for multilingual self-attentional translation models. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 261–271, Brussels, Belgium. Association for Computational Linguistics.

Shaomu Tan, Di Wu, and Christof Monz. 2024. Neuron specialization: Leveraging intrinsic task modularity for multilingual machine translation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6506–6527, Miami, Florida, USA. Association for Computational Linguistics.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, and 20 others. 2022. No language left behind: Scaling human-centered machine translation. *Preprint*, arXiv:2207.04672.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. 2024. Knowledge fusion of large language models. *Preprint*, arXiv:2401.10491.

Qian Wang and Jiajun Zhang. 2022. Parameter differentiation based multilingual neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11440–11448.

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.

Qiyu Wu, Masaaki Nagata, Zhongtao Miao, and Yoshimasa Tsuruoka. 2024. Word alignment as preference for machine translation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3223–3239, Miami, Florida, USA. Association for Computational Linguistics.

Chaojun Xiao, Zhengyan Zhang, Chenyang Song, Dazhi Jiang, Feng Yao, Xu Han, Xiaozhi Wang, Shuo Wang, Yufei Huang, Guanyu Lin, Yingfa Chen, Weilin Zhao, Yuge Tu, Zexuan Zhong, Ao Zhang, Chenglei Si, Khai Hao Moo, Chenyang Zhao, Huimin Chen, and 4 others. 2024. Configurable foundation models: Building llms from a modular perspective. *Preprint*, arXiv:2409.02877.

Haoran Xu, Young Jin Kim, Amr Sharaf, and Hany Hassan Awadalla. 2024a. A paradigm shift in machine translation: Boosting translation performance of large language models. In *The Twelfth International Conference on Learning Representations*.

Haoran Xu, Kenton Murray, Philipp Koehn, Hieu Hoang, Akiko Eriguchi, and Huda Khayrallah. 2025. X-ALMA: Plug & play modules and adaptive rejection for quality translation at scale. In *The Thirteenth International Conference on Learning Representations*.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. 2024b. Contrastive preference optimization: Pushing the boundaries of LLM performance in machine translation. In *Forty-first International Conference on Machine Learning*.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch. *Preprint*, arXiv:2311.03099.

Chengzhi Zhong, Qianying Liu, Fei Cheng, Junfeng Jiang, Zhen Wan, Chenhui Chu, Yugo Murawaki, and Sadao Kurohashi. 2025. What language do non-English-centric large language models think in? In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 26333–26346, Vienna, Austria. Association for Computational Linguistics.

Performance Level	Language	Code	$\rightarrow en$	$en \rightarrow$
High	French	fra	56.20	46.46
	Portuguese	por	58.35	47.51
	Spanish	spa	50.19	41.91
	German	deu	52.33	37.70
Medium	Czech	ces	38.45	20.94
	Asturian	ast	37.26	18.97
	Japanese	jpn	38.54	15.65
	Ukrainian	ukr	36.36	17.49
Low	Moroccan Tamazight	zgh	9.04	0.89
	Tamasheq	taq	9.56	1.04
	Acehnese	ace	13.69	0.70
	Minangkabau	min	13.94	0.74

Table 5: The selected 12 languages from FLORES+. We divide them into three performance levels (High, Medium, Low) based on their English-centric translation ability, measured by the average chrF++ scores when translating to English ($\rightarrow en$) and from English ($en \rightarrow$).

A Dataset Setting

We report chrF++ scores translating from ($en \rightarrow$) and to ($\rightarrow en$) of the 12 selected languages in Table 5. The 12 languages are grouped into three performance levels based on the average score s of these two directions: high-performance (≥ 40), medium-performance ($20 < s < 40$), low-performance (≤ 20). As shown in the results, for all languages, translating into English consistently yields higher chrF++ scores than translating from English. This also aligns with the general intuition that language understanding is much easier than language generation for LLMs.

B Training Details

For Qwen2.5-0.5B-Instruct, we use a learning rate of 2e-5 and train for 5 epochs, reporting the best chrF++ score as the full-parameter fine-tuning baseline. For English-centric LoRA training, we use the same learning rate, set the LoRA scaling factor $\alpha = 4r$, and train for 10 epochs.

For Qwen2.5-7B-Instruct, we use a learning rate of 1e-6 and train 10 epochs for the full-parameter fine-tuning baseline. We use a learning rate of 2e-5, set the LoRA scaling factor $\alpha = r$ and train 20 epochs for LoRA training.

For Llama3.1-8B-Instruct, we use a learning rate of 1e-6, train 10 epochs for the full-parameter fine-tuning baseline. For LoRA training, we notify that Llama needs a larger learning rate than Qwen during experiments; we use a learning rate of 1e-4, set the LoRA scaling factor $\alpha = r$, and train 20 epochs.

Consider the differing learning dynamics across languages, to avoid complex, language-specific hyperparameter searching, we adopt a small learning rate for all languages and extend training epochs. We then select the checkpoint that achieves the best performance on English-centric directions, ensuring that the appropriate LoRA module is selected for each language.

C Results on High- and Medium-performance Languages

We further extend our experiments to four high-performance languages (French, German, Portuguese, Spanish) and four medium-performance languages (Czech, Asturian, Japanese, Ukrainian), as shown in Table 6 and 7. For medium-performance languages, modular fine-tuning achieves performance comparable to traditional full-parameter fine-tuning, while using only LoRA$_{in}$ yields very limited improvement. For high-performance languages, we observe that modular fine-tuning even out-performs traditional full-parameter fine-tuning. We believe that this is because these languages are easier to overfit during training, leading to performance degradation. Additionally, incorporating LoRA$_{in}$ provides no benefit in this setting. These findings suggest that for high- and medium-performance languages, the model has already acquired sufficient knowledge during pre-training, making the use of LoRA$_{in}$ unnecessary.

D The Latent Language of Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct

Following Wendler et al. (2024), we visualize the latent probabilities of five languages using the logit lens: German (de), French (fr), Chinese (zh), English (en), and Russian (ru). As shown in Figure 4 and 5, we present the language probabilities across 12 translation directions using Llama3.1-8B-Insturct and Qwen2.5-7B-Instruct. Both Qwen and Llama exhibit a similar tendency that the input word is first transformed into Chinese and English in the middle layers and then converted into the target language in the final layers. For directions that do not involve Chinese, Llama shows a slightly stronger preference for English, while Chinese plays a dominant role in Qwen.

Methods	deu→	fra→	por→	spa→	→deu	→fra	→por	→spa	AVG
	Language Directions								
Pre-train	35.90	38.42	38.82	37.20	33.51	39.49	38.55	38.79	37.58
Full-parameter	38.11	39.54	39.51	37.80	33.77	40.75	41.16	39.28	38.74
Only InputLoRA(r=4)	33.07	40.07	40.11	37.95	32.55	39.70	39.77	39.20	37.80
Only OutputLoRA(r=4)	39.63	**42.83**	**43.43**	41.29	**35.75**	**44.43**	45.11	**41.91**	**41.80**
Merge(1:9,r=4)	39.51	42.50	43.14	41.26	35.52	44.23	<u>**45.27**</u>	41.38	41.60
Merge(3:7,r=4)	39.50	42.37	42.94	40.83	35.27	44.03	45.03	41.30	41.41
Merge(5:5,r=4)	39.27	42.19	42.40	40.16	34.60	43.61	44.79	41.02	41.00
Merge(7:3,r=4)	35.57	41.66	41.23	39.42	33.88	41.89	43.57	38.53	39.47
Merge(9:1,r=4)	30.19	40.15	39.32	37.99	32.56	38.72	39.94	36.43	36.91
Only InputLoRA(r=8)	29.31	39.80	39.61	36.43	32.24	37.06	37.74	38.12	36.29
Only OutputLoRA(r=8)	**39.71**	42.79	43.27	**41.30**	35.73	44.32	45.19	41.82	41.77
Merge(1:9,r=8)	39.36	42.57	43.18	41.21	35.63	44.25	45.12	41.33	41.58
Merge(3:7,r=8)	39.32	42.37	43.01	40.64	35.11	44.00	45.00	41.23	41.33
Merge(5:5,r=8)	38.80	42.16	42.68	39.89	34.38	43.51	44.69	40.95	40.88
Merge(7:3,r=8)	32.64	41.42	40.66	38.67	32.82	40.10	43.41	37.04	38.34
Merge(9:1,r=8)	25.68	39.69	38.12	36.16	31.19	35.50	38.61	34.35	34.91

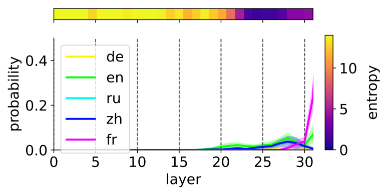
Table 6: The chrF++ scores for four high-performance languages on Qwen2.5-0.5B-Instruct. The merging ratio and the LoRA rank are denoted as w_{in}:w_{out} and r respectively. Scores where merging outperforms using only LoRA$_{in}$ and LoRA$_{out}$ are underlined.

E The Latent Language of Llama2-13B and Alma-13B

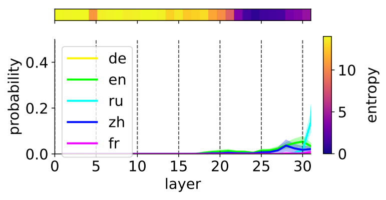
We present the language probabilities across 12 translation directions using Llama2-13B and Alma-13B in Figure 6 and 7. Alma is fine-tuned from Llama2 with billions of multilingual tokens. We find that there is no big difference between Llama and Alma, indicating that the multilingual fine-tuning will not influence the latent language of LLMs.

Methods	Language Directions								AVG
	ast→	ces→	jpn→	ukr→	→ast	→ces	→jpn	→ukr	
Pre-train	11.08	9.81	14.22	13.08	16.75	16.85	5.39	9.20	12.05
Full-parameter-94	**18.33**	**21.04**	**21.85**	**20.29**	27.82	**20.36**	12.09	**21.25**	**20.38**
Only InputLoRA(r=4)	14.77	14.46	12.92	14.21	17.27	16.88	9.32	12.90	14.09
Only OutputLoRA(r=4)	16.42	19.75	20.54	19.93	26.78	19.58	11.63	18.65	19.16
Merge(1:9,r=4)	$\underline{16.73}$	$\underline{19.94}$	20.14	$\underline{19.98}$	$\underline{26.88}$	$\underline{19.72}$	11.46	$\underline{18.73}$	$\underline{19.20}$
Merge(3:7,r=4)	$\underline{17.11}$	$\underline{20.15}$	19.99	$\underline{19.97}$	$\underline{27.36}$	19.53	$\underline{11.76}$	18.58	$\underline{19.31}$
Merge(5:5,r=4)	16.76	19.99	19.60	20.01	**27.88**	19.34	11.09	18.05	19.09
Merge(7:3,r=4)	16.21	19.35	17.21	18.94	$\underline{27.00}$	18.62	10.06	16.02	17.93
Merge(9:1,r=4)	14.80	17.63	13.38	16.20	23.37	17.10	8.27	13.26	15.50
Only InputLoRA(r=8)	15.08	13.88	12.48	13.06	16.99	16.48	8.54	12.49	13.63
Only OutputLoRA(r=8)	16.77	19.98	20.64	19.86	26.46	19.83	11.59	19.37	19.31
Merge(1:9,r=8)	$\underline{17.03}$	$\underline{20.07}$	20.22	$\underline{20.03}$	$\underline{26.49}$	$\underline{20.03}$	11.49	19.34	$\underline{19.34}$
Merge(3:7,r=8)	$\underline{17.27}$	$\underline{20.56}$	20.20	$\underline{20.14}$	$\underline{27.38}$	$\underline{19.92}$	$\underline{11.79}$	19.09	$\underline{19.54}$
Merge(5:5,r=8)	$\underline{17.10}$	$\underline{20.22}$	19.67	19.72	$\underline{27.82}$	19.42	11.30	18.18	19.18
Merge(7:3,r=8)	16.54	18.54	16.43	18.15	$\underline{26.88}$	18.17	9.89	14.72	17.42
Merge(9:1,r=8)	14.98	16.60	12.29	15.12	23.16	16.49	7.67	11.67	14.75

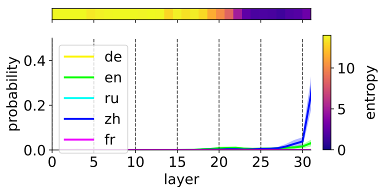
Table 7: The chrF++ scores for four medium-performance languages on Qwen2.5-0.5B-Instruct. The merging ratio and the LoRA rank are denoted as w_{in}:w_{out} and r respectively. Scores where merging outperforms using only LoRA$_{in}$ and LoRA$_{out}$ are underlined.



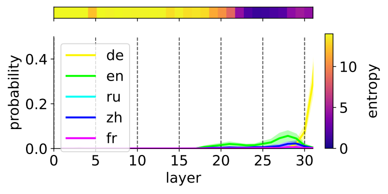
(a) de→fr



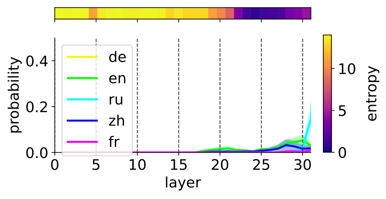
(b) de→ru



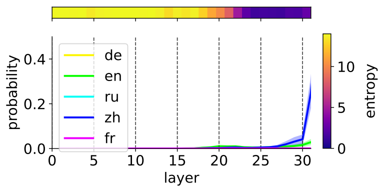
(c) de→zh



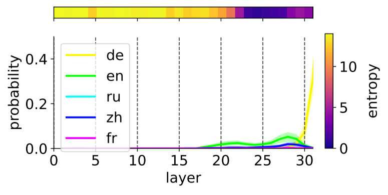
(d) fr→de



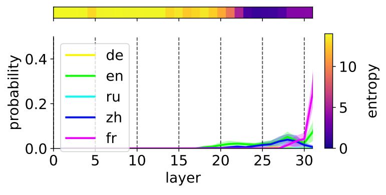
(e) fr→ru



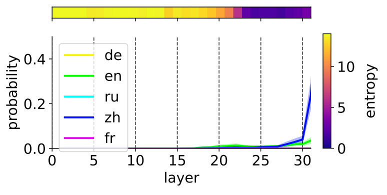
(f) fr→zh



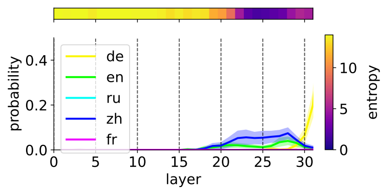
(g) ru→de



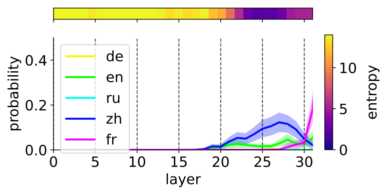
(h) ru→fr



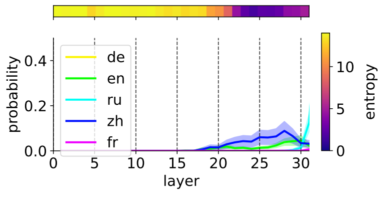
(i) ru→zh



(j) zh→de

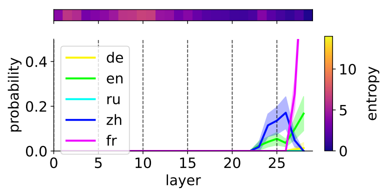


(k) zh→fr

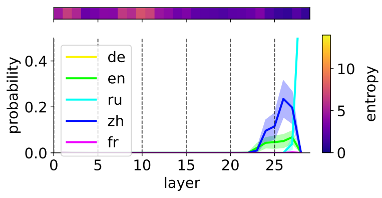


(l) zh→ru

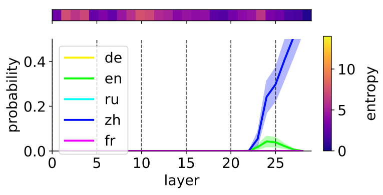
Figure 4: The language probabilities of 12 translation directions using Llama3.1-8B-Instruct. We show the probabilities on five languages at each layer: German (de), English (en), Russian (ru), Chinese (zh), French (fr). We also report the entropy on the top of the probabilities.



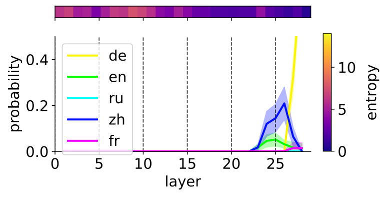
(a) de→fr



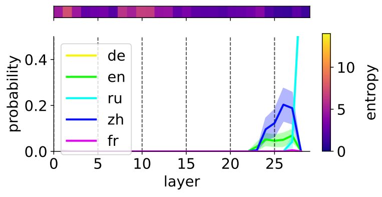
(b) de→ru



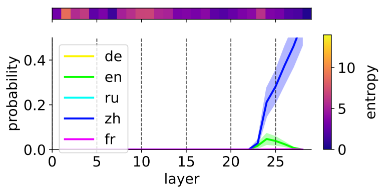
(c) de→zh



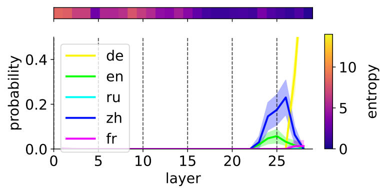
(d) fr→de



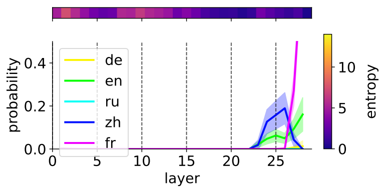
(e) fr→ru



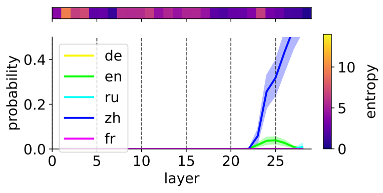
(f) fr→zh



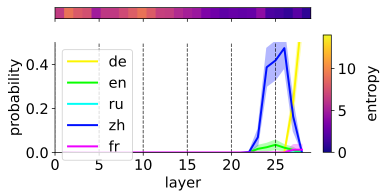
(g) ru→de



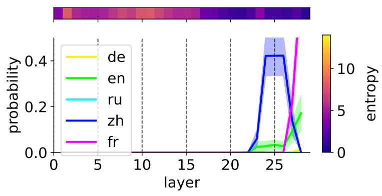
(h) ru→fr



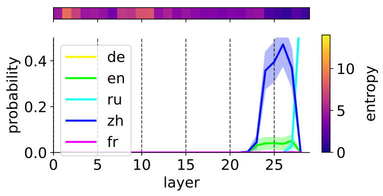
(i) ru→zh



(j) zh→de

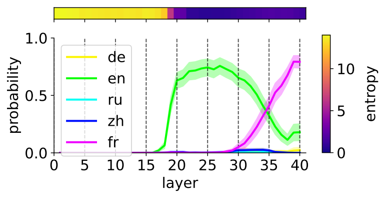


(k) zh→fr

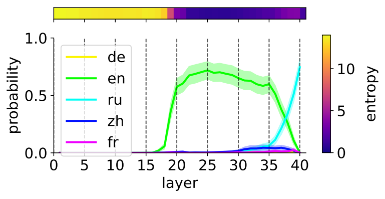


(l) zh→ru

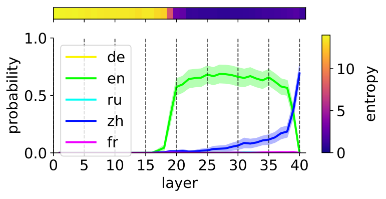
Figure 5: The language probabilities of 12 translation directions using Qwen2.5-7B-Instruct. We show the probabilities on five languages at each layer: German (de), English (en), Russian (ru), Chinese (zh), French (fr). We also report the entropy on the top of the probabilities.



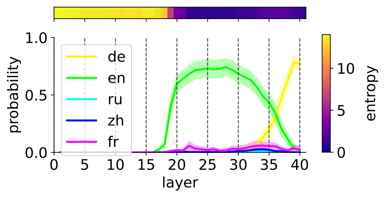
(a) de→fr



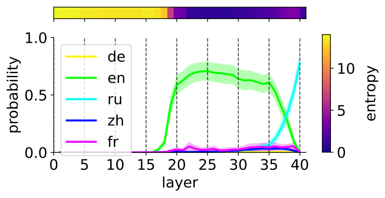
(b) de→ru



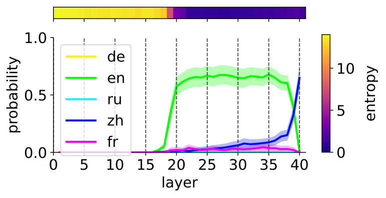
(c) de→zh



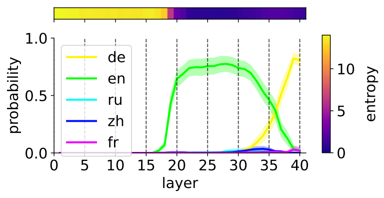
(d) fr→de



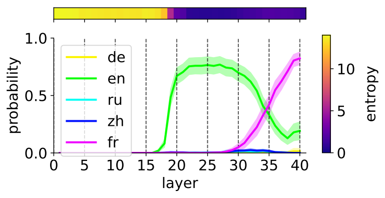
(e) fr→ru



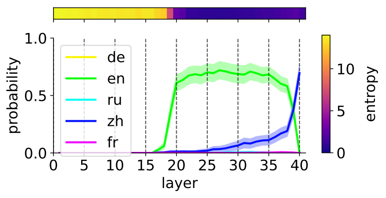
(f) fr→zh



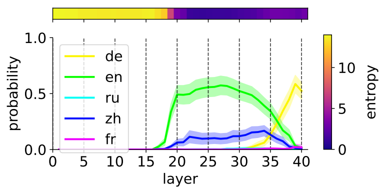
(g) ru→de



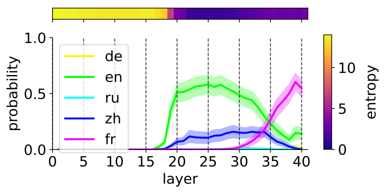
(h) ru→fr



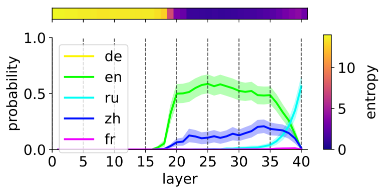
(i) ru→zh



(j) zh→de

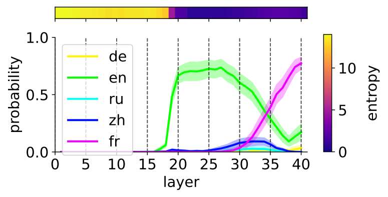


(k) zh→fr

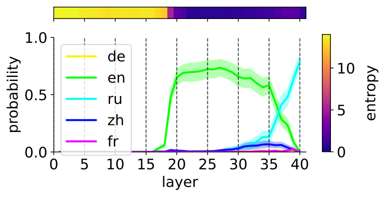


(l) zh→ru

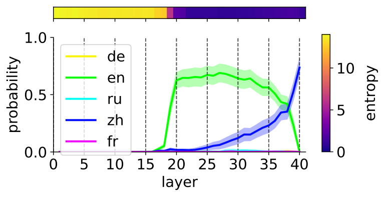
Figure 6: The language probabilities of 12 translation directions using Llama2-13B. We show the probabilities on five languages at each layer: German (de), English (en), Russian (ru), Chinese (zh), French (fr). We also report the entropy on the top of the probabilities.



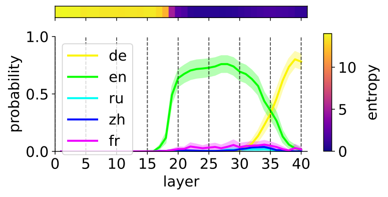
(a) de→fr



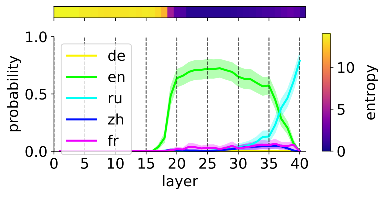
(b) de→ru



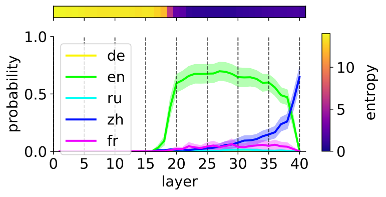
(c) de→zh



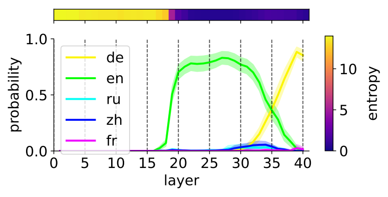
(d) fr→de



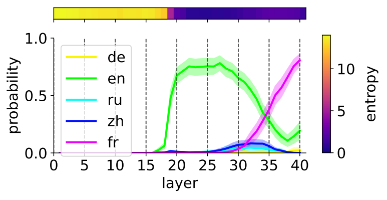
(e) fr→ru



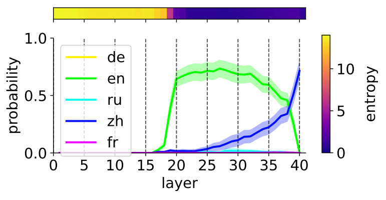
(f) fr→zh



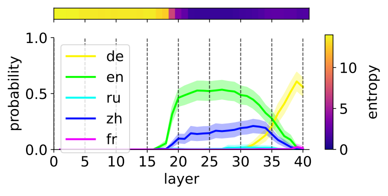
(g) ru→de



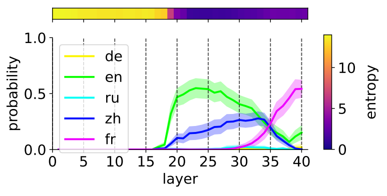
(h) ru→fr



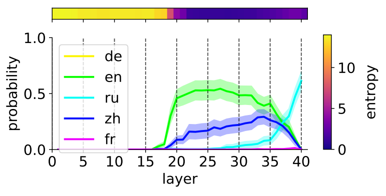
(i) ru→zh



(j) zh→de



(k) zh→fr



(l) zh→ru

Figure 7: The language probabilities of 12 translation directions using Alma-13B. We show the probabilities on five languages at each layer: German (de), English (en), Russian (ru), Chinese (zh), French (fr). We also report the entropy on the top of the probabilities.