Registering Source Tokens to Target Language Spaces in Multilingual Neural Machine Translation

Anonymous ACL submission

Abstract

The multilingual neural machine translation (MNMT) aims for arbitrary translations across multiple languages. Although MNMT-specific models trained by parallel data offer low costs in training and deployment, their performance consistently lags behind that of large language models (LLMs). In this work, we introduce **registering**, a novel method that enables a small MNMT-specific model to compete with LLMs. Specifically, we insert a set of artificial tokens specifying the target language, called registers, into the input sequence between the source and target tokens. By modifying the attention mask, the target token generation only pays attention to the activation of registers, representing the source tokens in the target language space. Experiments on EC-40, a large-scale benchmark, show that our method advances the state-of-the-art of MNMT. We further pretrain two models, namely MITRE (**m**ult**i**lingual **t**ranslation with **re**gisters), by 9.3 billion sentence pairs across 24 languages collected from public corpus. One of them, MITRE-913M, outperforms NLLB-3.3B, achieves comparable performance with commercial LLMs, and shows strong adaptability in fine-tuning. Finally, we open-source our models to facilitate further research and development in MNMT.

1 Introduction

Multilingual neural machine translation (MNMT) aims for arbitrary translations across multiple languages. Traditionally, training models specific to MNMT by using parallel data was highly appealing, not only because such MNMT-specific models can maintain a minimal number of parameters (Firat et al., 2016; Fan et al., 2020; NLLB Team, 2022), but also due to their potential for zero-shot translation, i.e., translating language pairs unseen during training, which helps address data scarcity in certain translation directions (Johnson et al., 2017; Fan et al., 2020; Zhang et al., 2020). However, the current mainstream solution for MNMT relies on large language models (LLMs), as the performance of MNMT-specific models has lagged behind that of LLMs (Zhu et al., 2023; Xu et al., 2024). Recent analyses (Chen et al., 2023; Tan and Monz, 2023) identify the off-target problem, i.e., translations fail to reach the intended target language, as a key factor causing the under-performance of MNMT-specific models. Moreover, Rios et al. (2020); Chen et al. (2023) show that constraining the target token generation into the target language space can alleviate the off-target problem.

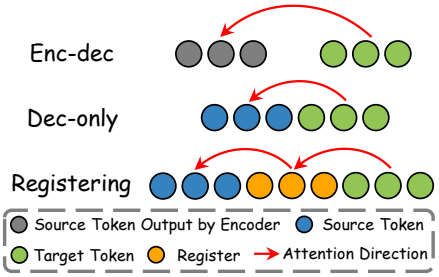


Figure 1: Illustration of the attention view among different architectures. "Token" refers to the representation corresponding to the token.

In this work, we present **registering**, a simple yet effective method designed for MNMT-specific models based on the decoder-only architecture (Dec-only) without introducing additional parameters. Specifically, we insert a set of artificial tokens between the source and target tokens, called registers, which indicate the target language only without any semantics. The registers are designed to be of the same length as the source tokens, because each register is expected to capture the semantics of its positionally-aligned source token and then represent it in the target language space. As illustrated in Figure 1, by modifying the attention mask, the generation of target tokens no longer follows the attention computation of encoder-decoder (Enc-Dec) and Dec-only architectures. Instead, it relies solely on the registers, located in the representational space of the intended target language.

We conduct two sets of experiments, evaluated with four automatic metrics: spBLEU (NLLB Team, 2022), chrF++ (Popović, 2015, 2017), COMET (Rei et al., 2020), and off-target ratio (Zhang et al., 2020). First, we experiment with EC-40 (Tan and Monz, 2023), a large-scale benchmark designed to assess zero-shot translation capability. Experimental results show that, compared to strong baselines, our method improves spBLEU scores by up to 71% on average across 1,640 directions with fewer parameters and drastically reduces the off-target ratio from 26.69% to 3.65%. Second, we collect 9.3 billion sentence pairs across 24 languages by sampling from the NLLB open dataset (NLLB Team, 2022) with the bridge language strategy (Fan et al., 2020). We then pre-train two models, MITRE-466M and MITRE-913M (**m**ult**i**lingual **t**ranslation with **re**gisters). One of them, MITRE-913M, not only outperforms NLLB-3.3B (NLLB Team, 2022) and GPT-3.5 Turbo (Brown et al., 2020) but also achieves competitive performance with GPT-4o mini (OpenAI, 2024). Also, we fine-tune the pre-trained models by full parameters and LoRA (Hu et al., 2022) in three distinct scenarios, demonstrating the superior adaptability of MITRE in fine-tuning. Finally, by analyzing the attention mechanisms and the representation distribution in translation instances at the token level, we confirm that the register mirrors the corresponding source token in the target language space.

2 Related Work

Johnson et al. (2017) proposed adding a language tag as a translation instruction at the beginning of tokens, marking the beginning of MNMT-specific models. Recent analyses (Chen et al., 2023; Tan and Monz, 2023) show that addressing the off-target problem is key to improving zero-shot translation. Early works (Rios et al., 2020; Qu and Watanabe, 2022) tried to use language-specific dictionaries or components to isolate generation across languages, but this was costly and hindered knowledge sharing (Zhang et al., 2021). As a compromise, Chen et al. (2023) proposed adding language-specific subsets to a shared dictionary to mitigate the off-target problem. Beyond explicitly adding language-specific parameters, optimizing internal representations implicitly improves zero-shot translation. Specifically, aligning semantic information across languages (Pan et al., 2021; Bu et al., 2024) and strengthening the translation instruction toward the target language (Stap et al., 2023; Qu et al., 2024a; Sun et al., 2024) help mitigate the off-target problem. Our proposed method is a combination of explicit and implicit strategies, as it separates generation-related representations by language.

In methodology, the translation instruction used in MNMT-specific models (Johnson et al., 2017) is similar to the artificial tokens in prefix-tuning (Li and Liang, 2021). In fact, we are methodologically inspired by gisting, a variation of prefix-tuning proposed by Mu et al. (2023). Specifically, they modified the attention mask to compress information into a set of artificial tokens, used in generation to eliminate the need for the original sequence. However, the difference in concept is that our proposed method, registering, aims to transfer each source token's semantics into the positionally-aligned register, which can be regarded a representation-level container pointing to the target language. In other words, registering is conceptually similar to chain-of-thought (Wei et al., 2023), where the process represents "rethinking" the source tokens from the perspective of the target language.

Additionally, given that LLMs already exhibit strong MNMT capabilities (Zhu et al., 2023), fine-tuning LLMs into MNMT-specific models has become a popular direction of exploration. However, the results in this direction are still limited, such as performance still falling behind commercial LLMs (Yang et al., 2023) and fewer supported languages e.g., 5 languages of Xu et al. (2024) and 10 languages of Alves et al. (2024). In this work, we demonstrate the potential of directly training an MNMT-specific model with parallel data only, aiming to drive further discussion on MNMT.

3 Multilingual Translation With Registers

3.1 Multilingual Neural Machine Translation

Given a multilingual corpus $\mathbb{C}$ spanning multiple translation directions, each instance in $\mathbb{C}$ is defined as $(\boldsymbol{x}, \boldsymbol{y})$, consisting of a set of source tokens $\boldsymbol{x} = x_1, \ldots, x_I$ and a set of target tokens $\boldsymbol{y} = y_1, \ldots, y_J$. Also, we introduce a set of language tags $\mathbb{L} = \{l_1, \ldots, l_K\}$, where language tags are artificial tokens individually corresponding to K languages in $\mathbb{C}$. Following Johnson et al. (2017); Wu et al. (2021), we add a tag indicating the language of $\boldsymbol{y}$ at the beginning of $\boldsymbol{x}$ as the translation instruction for multilingual neural machine translation (MNMT), denoted by $l_{\boldsymbol{y}}$. Consequently, the input fed into the model specific to MNMT be-

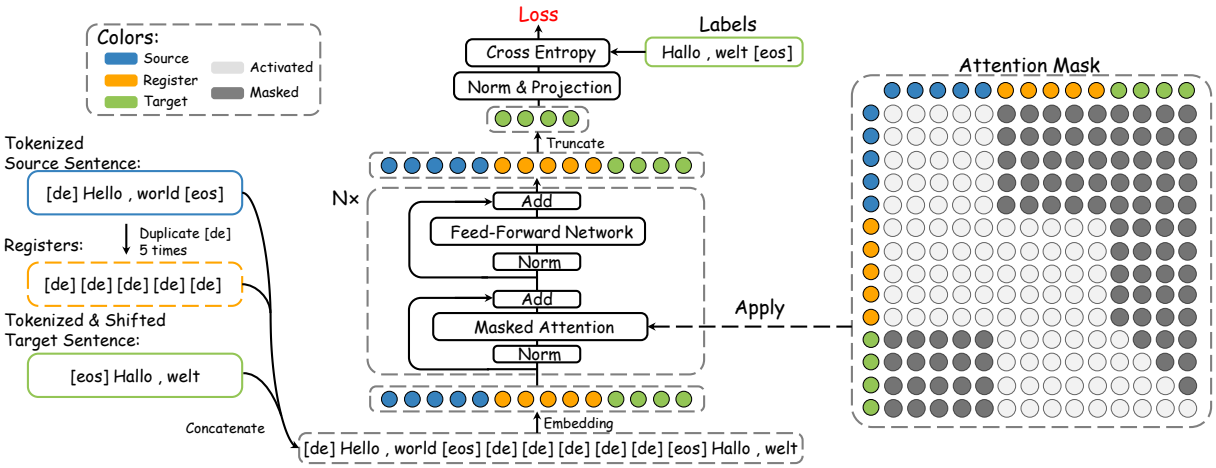


Figure 2: Illustration of registering. The example depicts a translation from English to German. The illustrated model stacks N layers, each following the Transformer decoder layer structure (Vaswani et al., 2017) with pre-normalization (Xiong et al., 2020). Notably, each circle represents a token and its representation in the generation.

comes $\boldsymbol{x}' = l_{\boldsymbol{y}}, x_1, \ldots, x_I$. Formally, we train the model over all instances of $\mathbb{C}$ by optimizing the following cross entropy loss:

$$\mathcal{L}_{\text{ce}} = -\sum_{\boldsymbol{x}',\boldsymbol{y}\in\mathbb{C}} \sum_{j=1}^{J} \log\ p(y_j \mid \boldsymbol{x}', \boldsymbol{y}_{<j}), \quad (1)$$

where $p(y_j \mid \boldsymbol{x}', \boldsymbol{y}_{<j})$ represents the probability distribution for generating y_j by MNMT-specific model. The current state-of-the-art models (Fan et al., 2020; NLLB Team, 2022) utilize the encoder-decoder architecture (Enc-dec), where the generation of y_j can be expressed as:

$$y_j = \text{decoder}(\text{encoder}(\boldsymbol{x}'), \boldsymbol{y}_{<j}). \quad (2)$$

Also, Gao et al. (2022); Zhang et al. (2022) show that the MNMT-specific model can be implemented with a decoder-only architecture (Dec-only). In this setup[1], the generation can be described as:

$$y_j = \text{decoder}(\boldsymbol{x}', \boldsymbol{y}_{<j}). \quad (3)$$

3.2 Registering

Gu et al. (2019); Qu et al. (2024b) suggest that $l_{\boldsymbol{y}}$ is a key factor causing the off-target problem. Specifically, although $l_{\boldsymbol{y}}$ is a translation instruction distinctly representing the target language, the generation cannot strictly depend on $l_{\boldsymbol{y}}$ due to the dilution of other source tokens in the attention mechanism. Therefore, registering is proposed to address this problem by constraining the generation within the target language space, where all registers used in generation have the same function with $l_{\boldsymbol{y}}$.

Given a Dec-only model[2], we begin by initializing a set of artificial tokens corresponding to the target language, denoted by $\boldsymbol{r} = r_1, \ldots, r_{I+1}$, matching the length of $\boldsymbol{x}'$. Notably, since $l_{\boldsymbol{y}}$ has the same function that we designed for r, $\boldsymbol{r}$ is initialized by duplicating $l_{\boldsymbol{y}}$. We then insert $\boldsymbol{r}$ into the input sequence between $\boldsymbol{x}'$ and $\boldsymbol{y}$, thus reformulating the generation process of Equation 3 as:

$$y_j = \text{decoder}(\boldsymbol{x}', \boldsymbol{r}, \boldsymbol{y}_{<j}). \quad (4)$$

The key step of registering is modifying the Transformer attention mask (Vaswani et al., 2017) to remove source tokens from the view of target tokens. As shown in Figure 2, we initialize the attention mask from the prefix Dec-only scheme (Dong et al., 2019), where the source tokens compute attention for each other bidirectionally, while the target tokens only compute attention for the previous ones. We then adjust the mask to control token-level representation according to the following rules: (1) $\boldsymbol{r}$ pays attention to $\boldsymbol{x}'$; (2) $\boldsymbol{r}$ computes attention bidirectionally for $\boldsymbol{r}$; (3) y_j pays attention to $\boldsymbol{r}$ and $\boldsymbol{y}_{<j}$. With this design, the generation of $\boldsymbol{y}$ can solely rely on the activation of $\boldsymbol{r}$, where the activation of r_i not only functions as a representational container of the target language but also carries the semantics of x_i. As a result, the generation is strictly constrained within the target language space to effectively address the off-target problem.

[1]We follow Gao et al. (2022) to train a Dec-only MNMT-specific model with Equation 1 rather than the language modeling loss (Radford et al., 2018).

[2]Intuitively, Dec-only is more parameter-efficient than Enc-dec, as separate components in the latter process source and target tokens. Therefore, given that registering can address the off-target problem, Dec-only is the preferable choice. Supporting experiments are provided in Appendix J.

			High		Med		Low		Extra Low					
	#params	Method	→	←	→	←	→	←	→	←	sup.	zero.	avg.	off.(%)
Enc-dec	242M	vanilla	9.46	11.05	7.49	9.80	5.03	3.95	5.41	2.59	29.06	5.86	6.99	48.40
		+CL	14.21	14.19	12.19	14.18	7.89	7.55	7.66	6.04	29.03	9.74	10.68	19.08
		+LCS	10.44	13.67	9.34	13.17	8.73	6.07	8.49	4.10	**29.18**	8.35	9.37	22.43
	259M	+LAVS	9.71	12.22	7.74	11.19	5.71	3.98	6.37	2.99	29.07	6.40	7.54	42.98
Dec-only	217M	vanilla	11.57	11.51	9.48	11.03	6.01	6.06	5.74	4.20	28.61	7.30	8.34	22.21
		+TDO	13.33	13.50	10.53	12.75	7.13	6.88	6.74	4.60	28.84	8.62	9.61	27.18
		+Ours	**15.43**	**15.46**	**13.88**	**14.62**	**8.94**	**8.99**	**8.76**	**7.94**	28.90	**11.05**	**11.92**	**4.65**
Enc-dec	418M	vanilla	12.66	15.02	10.86	14.50	7.40	5.04	7.12	3.47	30.28	8.64	9.69	26.69
		+CL	15.89	15.97	13.67	16.15	8.36	8.16	8.32	5.96	**30.54**	10.79	11.76	19.99
		+LCS	10.79	16.19	10.00	15.31	9.99	5.39	9.41	3.70	30.33	9.25	10.28	23.47
	430M	+LAVS	14.03	16.39	12.50	16.31	8.61	6.31	8.45	3.57	30.20	10.03	11.01	21.73
Dec-only	368M	vanilla	14.37	15.07	12.25	15.02	8.27	7.40	7.71	5.11	29.97	9.84	10.82	19.01
		+TDO	15.27	15.79	12.83	15.56	8.44	7.96	8.15	5.40	30.23	10.40	11.37	23.14
		+Ours	**16.81**	**16.98**	**15.25**	**16.57**	**10.10**	**9.88**	**9.64**	**8.37**	29.88	**12.26**	**13.12**	**3.65**

Table 1: Averaged spBLEU scores of results on EC-40, the last column (off.) lists the off-target ratio averaged from all directions, the scores of chrF++ and COMET are reported in Tables 8 and 9, as discussed in Appendix H. We report scores by grouping languages that have the same resource tier. Then, → includes directions translating from the corresponding group to languages out of this group, and ← includes directions translating to the corresponding group. sup., zero, and avg. abbreviate the average of supervised translations, the average of zero-shot translations, and the average of all translations, respectively. The best score in each column of a block is in bold.

4 Experiment: On Benchmark

4.1 Dataset and Evaluation

We conduct the first set of experiments on a large zero-shot translation benchmark, EC-40 (Tan and Monz, 2023), consisting of 120 million translation instances spanning 41 languages from five language families in the training data[3]. Except for English, each family includes eight languages, categorized into four resource tiers, namely, High, Medium, Low, and Extra Low, corresponding to 5 million, 1 million, 100,000, and 50,000 sentence pairs, respectively. For testing the zero-shot translation capability, all training directions in EC-40 involve English, either as the source or target language, resulting in 80 supervised directions. Then, we evaluate the trained models on all supervised and zero-shot directions, i.e., 1,640 directions. Different from the original setup in Tan and Monz (2023), we follow NLLB Team (2022); Cao et al. (2024) to standardize the validation and testing processes using Flores[4], which is a high-quality parallel dataset available for over 200 languages. Specifically, we use the *dev* and *devtest* sets of Flores for validation and testing, containing 997 and 1,012 sentences per language, respectively.

In the evaluation, we set the beam size to 5 in inference. We employ four automatic metrics to evaluate inference results on the test set: (1) spBLEU (NLLB Team, 2022), a variant of BLEU (Papineni et al., 2002; Post, 2018) used for Flores, unifies tokenization across languages through an open-source tokenizer[5]; (2) chrF++ (Popović, 2015, 2017) assesses character-level overlap and balances precision with recall; (3) COMET[6] (Rei et al., 2020) evaluates quality by comparing generated translations, reference translations, and source sentences at a representation level; (4) we report the off-target ratio (Zhang et al., 2020) as a supplementary metric, because the testing tool[7] is not absolutely accurate as it relies on recognizing language-specific tokens. Since COMET and off-target ratio evaluations lack support for certain languages, we compute these scores only for supported languages, as listed in Appendix G.

4.2 Configuration and Baseline

The modeling follows the manner of Transformer (Vaswani et al., 2017) with an embedding size of 1,024, an inner size of 4,096, and 16 attention heads. We divide the models into two configurations based on model depth. we first stack 12 layers in modeling and balance the number of layers between the encoder and decoder in Enc-dec, resulting in 242M parameters for Enc-dec and 217M parameters for Dec-only. Then, models include 24 layers in the

[3]Details of EC-40 are described in Appendix A.

[4]https://github.com/openlanguagedata/flores.

[5]https://tinyurl.com/flores200sacrebleuspm.

[6]All COMET scores are computed using *Unbabel/wmt22-comet-da* (Rei et al., 2022).

[7]https://github.com/LlmKira/fast-langdetect.

second configuration, yielding 418M parameters for Enc-dec and 368M parameters for Dec-only.

Apart from the vanilla Enc-dec and Dec-only, we also reproduce four related methods mentioned in Section 2 as baselines: (1) LAVS (Chen et al., 2023) adds language-specific tokens to the shared dictionary; (2) CL (Pan et al., 2021) aligns sentence-level semantic representations across languages using the encoder output; (3) LCS (Sun et al., 2024) strengthens translation instructions for Enc-dec by biasing token representations in the encoder with a target language embedding. This mechanism resembles $\boldsymbol{r}$ but uses a different operation; (4) TDO (Qu et al., 2024b) strengthens translation instructions for Dec-only by dividing the process of Dec-only into two phases, specifically, encoding source tokens with stronger target language features at the first phase, and then, concatenating the encoded source tokens and target tokens to feed into Dec-only models. We list all implementation and training details of these baselines in Appendix C.

4.3 Result

Experimental results shown in Table 1 exhibit consistent trends across both configurations, where our method consistently performs the best. The most notable improvement is in the off-target ratio, where the metric reduces from 48.40% to 4.65% in 12-layer models and from 26.69% to 3.65% in 24-layer models. These results indicate that registering nearly resolves the off-target problem. Moreover, although our method does not achieve the highest performance in supervised translation, this phenomenon is not a weakness, because the higher supervised performance of vanilla models is attributed to the overfitting (Gu et al., 2019; Liu et al., 2021). Additionally, we observe that registering significantly outperforms LAVS. Based on our discussion of LAVS and its underlying methods in Section 2, we state that simply adding language-specific parameters is not a cost-efficient solution.

We also observe that the gains of spBLEU scores from related methods tend to diminish as model parameters increase. In the 12-layer models, the highest gain among four related methods over vanilla models in zero-shot translation is 3.88. In 24-layer models, the improvement decreases to 2.15. However, our method achieves more consistent improvements with gains of 5.19 and 3.62 in 12-layer and 24-layer models, respectively. From this comparison, we can conclude that registering demonstrates superior scalability.

Family (Group)	Languages	Bridge
English*	en	en
Germanic	de, nl, sv, da, af	de, nl
Romance	fr, es, it, pt, ro	fr, es
Slavic	ru, cs, pl, bg, uk	ru, cs
Malayo-Polynesian	id, ms, jv, tl	id, ms
Asian*	ja, zh, ko, vi	ja, zh

Table 2: Languages in data collection. Languages are shown by their ISO 639-1 codes. Decoration with * indicates a language group instead of a language family.

5 Experiment: Pre-trained Models

5.1 Data Collection with Bridge Languages

A robust and practical MNMT-specific model expects training on a dataset supervised across multiple translation directions rather than being English-centric (Zhang et al., 2020; Eriguchi et al., 2022). However, collecting data for every possible translation direction is infeasible, as the number of directions grows exponentially with the number of supported languages. In this work, limited by our computational resources, we adopt the Bridge Language strategy (Fan et al., 2020) to collect data across 24 languages spanning more than five language families.

As shown in Table 2, we group languages by family except for English. The Germanic, Romance, and Slavic groups belong to European language families, and the Malayo-Polynesian differs significantly from these European languages. In addition, we define a special group, Asian, which includes four languages predominantly used in the Asian continent: ja, zh, ko, and vi. While these languages belong to different families, they share certain similarities due to their geographic proximity. We designate the two most resource-rich languages in each group as bridge languages and follow these rules for data collection: (1) en connects with all languages; (2) bridge languages connect with each other; (3) bridge languages connect with the remaining languages within their respective groups. Given that ms cannot meet rules (2) and (3) due to its low resource, we collect additional data for ms where possible. Based on the above strategy, out of 552 possible translation directions, we collect data from the reproduced version of the NLLB dataset[8] (NLLB Team, 2022) for a total of 194 directions, resulting in 9.3 billion sentence pairs for our pre-training.[9]

[8] https://opus.nlpl.eu/NLLB/corpus/version/NLLB

[9] Appendix B reports the data distribution at the language-family and language level.

Model		English		Germanic		Romance		Slavic		Mal.-Polyn.		Asian		
		→	←	→	←	→	←	→	←	→	←	→	←	avg.
M2M	483M	30.36	31.92	24.40	22.58	24.01	25.81	22.59	23.40	17.94	16.50	18.30	18.37	22.10
	615M	30.69	31.98	26.35	25.56	25.47	27.52	24.02	24.77	20.11	17.49	19.31	19.09	23.65
	1.2B	35.92	35.14	29.51	26.82	28.40	28.38	26.58	26.19	18.09	17.57	15.48	20.07	24.69
NLLB	615M	35.85	41.04	28.13	27.41	27.46	29.09	25.40	25.33	25.39	24.35	20.72	19.42	26.05
	1.3B	38.08	43.42	30.52	30.17	29.63	31.42	27.84	28.25	28.08	26.87	23.50	21.06	28.51
	3.3B	39.80	**45.08**	31.93	31.77	30.88	32.62	29.29	30.13	29.81	**28.08**	25.18	22.56	30.01
GPT	3.5 turbo	38.27	42.37	31.01	31.02	30.09	32.73	28.56	27.85	26.75	22.81	23.61	24.08	28.66
	4o mini	**41.49**	43.97	33.09	31.92	31.40	34.03	30.54	30.69	**31.01**	27.20	**26.34**	**27.53**	31.09
MITRE	466M	40.20	42.60	32.14	31.51	31.32	33.26	29.36	29.80	28.46	26.16	24.05	23.56	29.77
	913M	41.16	44.17	**33.34**	**32.95**	**32.53**	**34.23**	**30.74**	**31.26**	29.90	27.22	25.93	25.58	**31.15**

Table 3: Averaged spBLEU scores comparing MITRE with baselines. The off-target ratio is not reported due to the near-zero values in these large-scale models. chrF++ and COMET scores are provided in Tables 10 and 11, as discussed in Appendix I. Mal.-Polyn. abbreviates Malayo-Polynesian, and other abbreviations follow Table 1. Prompts used for GPT are reported in Appendix F. Additionally, we use green boxes to highlight scores exceeding NLLB-3.3B and blue boxes for those surpassing GPT-4o mini, where blue box has the priority.

5.2 Configuration and Baseline

We begin by training a vocabulary of 160,000 tokens using SentencePiece (Kudo and Richardson, 2018) on 150 million sentences randomly sampled from the training set. We then pre-train two models, named MITRE (**m**ul**ti**lingual **t**ranslation with **re**gisters), on 80 V100 GPUs with 466 million and 913 million parameters, respectively. We report complete details of modeling and training in Appendix D. The validation and testing process aligns with Section 4.1.

We compare our model against not only state-of-the-art MNMT-specific models, but also commercial LLMs, because commercial LLMs present the upper limit of fine-tuning open-sourced LLMs into MNMT-specific models (Section 2). First, the MNMT models include three versions of M2M (483M[10], 615M, and 1.2B) (Fan et al., 2020) and three versions of NLLB (615M-distilled, 1.3B, and 3.3B) (NLLB Team, 2022). Also, we include commercial LLMs, GPT-3.5 Turbo[11] (Brown et al., 2020) and GPT-4o mini[12] (OpenAI, 2024). Meanwhile, we include these NLLB models as baselines[13] in our fine-tuning experiments. Specifically, we create three scenarios randomly selecting 5, 25, and 100 translation directions from the possible directions. Then, we perform fine-tuning with full parameters and with LoRA (Hu et al., 2022) on the Flores *dev*, which contains 997 sentence pairs per direction. Fine-tuning settings are provided in Appendix E.

[10]The official name is M2M-418M, however, this model actually has 483M parameters.

[11]Version is *gpt-3.5-turbo-0125*.

[12]Version is *gpt-4o-mini-2024-07-18*.

[13]Due to the limitation of computational resources, NLLB-3.3B is not included in fine-tuning with full parameters.

5.3 Main Results

The experimental results comparing MITRE with baselines are shown in Table 3. We observe that although NLLB-3.3B surpasses MITRE-913M by 0.91 spBLEU scores for translations into English and by 0.86 scores for Malayo-Polynesian languages, MITRE-913M consistently achieves higher scores in other translation directions, with an overall average gain of 1.14 scores. Given that NLLB even surpasses GPT-4o mini by 1.11 scores in English translation, we infer that MITRE, a Dec-only model with registering, demonstrates better generalization than NLLB, based on Enc-dec. Notably, scaling parameters of NLLB from 1.3B to 3.3B yields only a gain of 1.50 scores, while MITRE attains a comparable gain of 1.38 scores with approximately 450M additional parameters. Furthermore, the alignment of training and validation loss for two MITRE models (Appendix D) reinforces our conclusion in Section 4.3 that registering provides superior scalability. Finally, based on all experimental results, we conclude that MITRE-466M performs competitively with NLLB-3.3B, while MITRE-913M not only outperforms NLLB-3.3B but also can compete with GPT-4o mini[14], showing the practical potential of our models.

5.4 Fine-tuning Results

Table 4 shows the fine-tuning results. By comparing NLLB and MITRE, we observe that MITRE outperforms NLLB in both scenarios: fine-tuning on a few translation directions and fine-tuning on

[14]Table 11 shows that the COMET scores of MITRE-913M are lower than GPT-4o mini. Therefore, although spBLEU and chrF++ scores of MITRE-913M are higher, we only claim that MITRE-913M performs competitively with GPT-4o mini. We provide an additional discussion in Appendix I.

model		5-direction spB.	5-direction com.	25-direction spB.	25-direction com.	100-direction spB.	100-direction com.
N.-615M	pre.	24.00	82.91	25.88	83.71	25.37	83.35
	lora	24.68	83.23	26.84	84.10	26.41	84.00
	f.t.	25.59	83.80	27.32	84.29	26.70	84.17
N.-1.3B	pre.	26.59	84.86	28.33	85.41	27.82	85.17
	lora	27.39	85.20	29.49	85.87	29.18	85.87
	f.t.	28.50	85.78	30.13	86.09	29.61	86.05
N.-3.3B	pre.	27.95	85.60	29.70	86.16	29.31	85.98
	lora	29.05	86.08	31.23	86.63	30.98	86.71
M.-466M	pre.	24.51	83.73	28.71	85.41	29.07	85.26
	lora	26.37	84.47	29.97	86.09	30.42	86.27
	f.t.	28.19	85.28	30.61	86.36	30.81	86.45
M.-913M	pre.	25.52	84.56	29.95	86.07	30.37	85.92
	lora	28.14	85.59	31.68	86.90	32.33	87.15
	f.t.	**30.09**	**86.45**	**32.47**	**87.23**	**32.73**	**87.35**

Table 4: Averaged spBLEU and COMET scores of results on three fine-tuning scenarios, where the specific translation directions are listed in Appendix B. N., M., pre., and f.t. abbreviate NLLB, MITRE, pre-trained models, and fine-tuning with full parameters, respectively. The best score is in bold, blue boxes highlights the largest gain in f.t. relative to pre., and green boxes highlights the largest gain in lora.

#layer	register	mask	spB.↑	chrf↑	com.↑	off.↓
12	✘	✘	8.34	22.34	55.71	22.21
	✔	✘	8.19	22.96	55.72	32.11
	✔	✔	11.92	29.02	61.19	4.65
24	✘	✘	10.82	26.04	59.95	19.01
	✔	✘	8.91	24.06	57.60	34.22
	✔	✔	13.12	30.51	63.54	3.65

Table 5: Averaged scores of ablation study on EC-40. Here, register means adding registers, and mask means modifying the attention mask.

multiple directions simultaneously. Specifically, we find that performance gains from fine-tuning increase with model size, and our MITRE-913M shows the highest improvement in both full parameter fine-tuning and LoRA-based fine-tuning. Additionally, since pre-trained models of both NLLB and MITRE achieve near-zero off-target ratios, these gains can be attributed to increased quality rather than addressing the off-target problem. This suggests that MITRE has a higher performance ceiling, likely due to our cost-effective data collection strategy, which may have constrained MITRE from reaching its theoretical maximum. Therefore, due to MITRE's superior fine-tuning capability, we reaffirm that its practical potential is remarkable.

6 Discussion

6.1 Ablation Study

We conduct two ablation studies to measure the impact of registering. First, we decompose registering into two steps: (1) adding registers and (2) modifying the attention mask. As shown in Table 5, merely adding registers reduces the performance of vanilla Dec-only models; registering only becomes effective after modifying the attention mask. This result aligns with expectations, i.e., the model without constraints on generation defaults to relying directly on source tokens instead of registers.

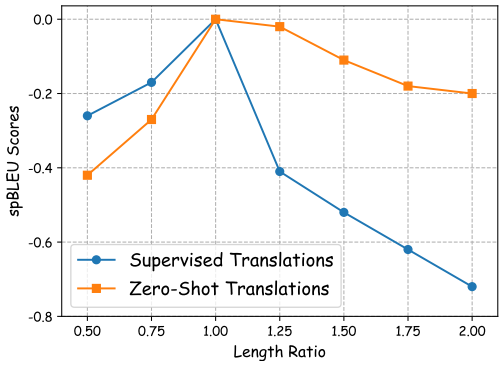


Figure 3: The spBLEU score variations on EC-40 with different length ratios of $\boldsymbol{r}$ to $\boldsymbol{x}'$, where only the length of $\boldsymbol{r}$ is changed and $\boldsymbol{x}'$ is fixed.

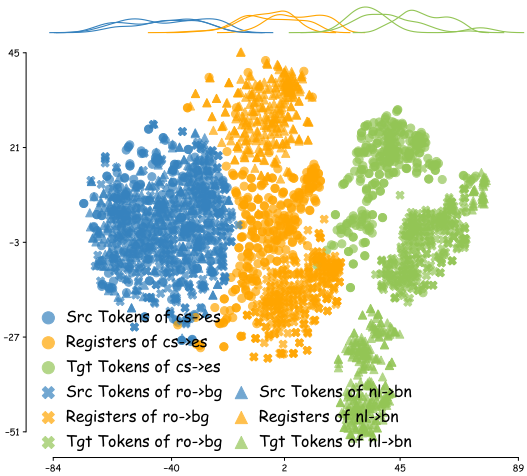
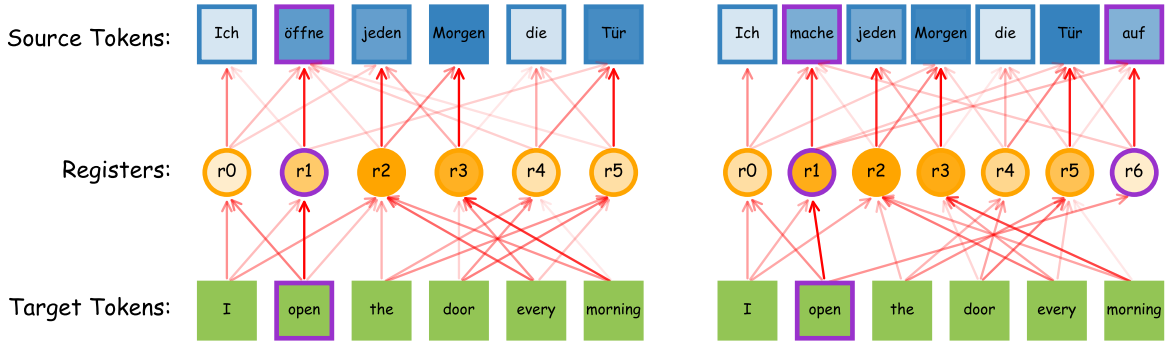


Figure 4: 2D distribution of token-level representations extracted from the output of the 24th layer of a model trained on EC-40. Each class listed in the legend contains 300 randomly sampled tokens. Appendix L shows the representational distributions from other layers.

In Section 3.2, we introduce that the lengths of $\boldsymbol{r}$ and $\boldsymbol{x}$ are matched to ensure a one-to-one correspondence between registers and source tokens. To validate this design, we vary the length of $\boldsymbol{r}$ while keeping $\boldsymbol{x}'$ fixed to observe performance trends. Specifically, a ratio less than 1.0 means that $\boldsymbol{r}$ augments $\boldsymbol{x}'$, while a ratio greater than 1.0 means $\boldsymbol{r}$ compresses $\boldsymbol{x}'$. The trend illustrated in Figure 3, where the length ratio of 1.0 is the optimal case, empirically supports our design.

6.2 Mechanism: registering source tokens to target language spaces

We analyze and validate the registering mechanism from two points: (1) the representation of registers



(a) Inseparable Verb (b) Separable Verb

Figure 5: Token-level attention weights illustration, where the weight of each token is averaged across all heads of a model trained on EC-40. 5a and 5b illustrate two instances translated from de to en. The top-3 attention directions for each token are labeled, with darker colors indicating higher attention weights. Note that while the target tokens for these two instances are identical, their source tokens are not, because the verbs in 5a and 5b are semantically equivalent but have different forms. To aid understanding, we highlight the verbs with purple borders: "öffne" in 5a and "mache ... auf" in 5b both correspond to the target verb "open". Then, the registers with the highest attention weights associated with these verbs are also marked with purple borders.

is located in the intended target language space, and (2) the register carries the semantics of the positionally-aligned source token.

We analyze token representations by randomly selecting 100 translation instances from three translation directions and applying t-SNE (van der Maaten and Hinton, 2008) to reduce them to two dimensions. Figure 4 shows the representation distribution in the final layer. First, source tokens, registers, and target tokens are clearly separated into distinct spaces, indicating that the model can distinguish their different functions. Additionally, source token representations from different languages cluster together, suggesting that the model processes them in a language-agnostic manner. Most importantly, for registers and target tokens, token representations for the three translation directions are clustered in separate spaces. This supports our design, where the register representation is located in the intended target language space.

The relationship between source tokens and registers can be exhibited by analyzing the attention weights in the generation based on a simple and interesting grammar of de. Figure 5 shows two translation instances translated from de to en, where they have the same semantics. We observe that the target tokens in both examples are identical, while their source tokens differ only in the verb form (highlighted with a purple border). Specifically, in Figure 5a, the verb is a single token, "öffne", whereas in Figure 5b, it consists of two distant tokens, "mache ... auf". Despite this difference, "öffne" and "mache ... auf" share the same semantics and can be replaced by each other. Figure 5 presents the attention weights for each token representation. In Figure 5a, the highest attention weight of the verb in the target tokens, i.e., "open", comes from r_1; in Figure 5b, "open" pays the highest attention weights to r_1 and r_6. Meanwhile, r_1 in Figure 5a aligns positionally with "öffne", and in Figure 5b, r_1 and r_6 align positionally with "mache ... auf". Additionally, we can observe that the highest attention weight of a register always comes from its positionally-aligned source token[15]. Based on the above, we conclude that the mechanism of registering aligns with our design, namely, the register's activation represents the target language and carries the semantics of the positionally-aligned token.

7 Conclusion

In this work, we present registering to address the off-target problem for MNMT-specific models. By introducing registers and modifying the attention mask, our method ensures that the generation of target tokens depends solely on the activation of registers. Analysis experiments demonstrate that the activation of registers carries the semantics of source tokens within the target language spaces. Using this method, we develop and open-source two MNMT-specific models, MITRE-466M and MITRE-913M, supporting translation across 24 languages. Experimental results show that MITRE performs competitively with commercial LLMs, setting a new state-of-the-art in MNMT.

[15] Appendix K shows more examples in Russian, Chinese, and Japanese, where all instances follow this statement.

8 Limitation

A key concern is our limited computational resources. Given that the training of MITRE-913M has already required 80 Tesla V100 GPUs for one month, MITRE supports 24 languages and we cannot further increase the supported languages. Although this number is far greater than the latest research in the community, e.g., 10 languages of Alves et al. (2024) and 5 languages of Xu et al. (2024), this number is fewer than supported languages of M2M, NLLB, and commercial LLMs. However, the comparison in Section 5 is relatively fair. Specifically (using NLLB as an example), first, our training data is collected from the reproduced version of the NLLB dataset, which includes fewer samples per translation direction than those used in training NLLB models. Second, as described in Section 5.1, our Bridge Language strategy results in fewer supervised translation directions, whereas NLLB is trained on as many directions as possible. Moreover, NLLB incorporates additional engineering strategies, e.g., back-translation (Edunov et al., 2018) and distillation (Hinton et al., 2015), whereas MITRE only iterates over the training set. Also, we directly compare MITRE-466M and MITRE-913M with NLLB-3.3B, where the parameter size difference helps offset the disparity in supported languages. Finally, we conduct fine-tuning experiments to compare MITRE and NLLB with the same settings.

Another limitation of our approach is the additional computational cost introduced by registers, as they double the number of source tokens. Based on our measurements on EC-40 using a Tesla V100 GPU, the training time for models with registers is 1.34 times that of the vanilla decoder-only model, 1.63 times that of the vanilla encoder-decoder model, and approximately equivalent (1.01 times) to the previous state-of-the-art method, CL (Pan et al., 2021). During inference, thanks to the KV Cache (refer to incremental decoding in Fairseq), the model with registers merely incurs a 1.1 times increase in computational cost compared to the vanilla decoder-only model. However, in practical usage, i.e., MITRE, the cost time of inference is substantially lower than that of NLLB and LLMs due to the smaller number of parameters.

9 Ethical Consideration

Although our training data is collected from public datasets, MITRE has not been evaluated for toxicity or undergone detoxification. Thus, while we open-source MITRE, we recommend its use primarily for research purposes or in applications only after thorough appropriate processing.

References

Duarte M. Alves, José Pombal, Nuno M. Guerreiro, Pedro H. Martins, João Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, Sweta Agrawal, Pierre Colombo, José G. C. de Souza, and André F. T. Martins. 2024. Tower: An open multilingual large language model for translation-related tasks. *Preprint*, arXiv:2402.17733.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *Preprint*, arXiv:1907.05019.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Mengyu Bu, Shuhao Gu, and Yang Feng. 2024. Improving multilingual neural machine translation by utilizing semantic and linguistic features. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 10410–10423, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Zhe Cao, Zhi Qu, Hidetaka Kamigaito, and Taro Watanabe. 2024. Exploring intrinsic language-specific subspaces in fine-tuning multilingual neural machine translation. *Preprint*, arXiv:2409.05224.

Liang Chen, Shuming Ma, Dongdong Zhang, Furu Wei, and Baobao Chang. 2023. On the off-target problem of zero-shot multilingual neural machine translation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 9542–9558, Toronto, Canada. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *Preprint*, arXiv:1905.03197.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*,

pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Akiko Eriguchi, Shufang Xie, Tao Qin, and Hany Hassan Awadalla. 2022. Building multilingual machine translation systems that serve arbitrary x-y translations. *Preprint*, arXiv:2206.14982.

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.

Orhan Firat, Baskaran Sankaran, Yaser Al-onaizan, Fatos T. Yarman Vural, and Kyunghyun Cho. 2016. Zero-resource translation with multi-lingual neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 268–277, Austin, Texas. Association for Computational Linguistics.

Yingbo Gao, Christian Herold, Zijian Yang, and Hermann Ney. 2022. Is encoder-decoder redundant for neural machine translation? In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 562–574, Online only. Association for Computational Linguistics.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O.K. Li. 2019. Improved zero-shot neural machine translation via ignoring spurious correlations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1258–1268, Florence, Italy. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *Preprint*, arXiv:1503.02531.

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*.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *Preprint*, arXiv:1412.6980.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597, Online. Association for Computational Linguistics.

Danni Liu, Jan Niehues, James Cross, Francisco Guzmán, and Xian Li. 2021. Improving zero-shot translation by disentangling positional information. 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 1259–1273, Online. Association for Computational Linguistics.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. 2023. Learning to compress prompts with gist tokens. In *Thirty-seventh Conference on Neural Information Processing Systems*.

NLLB Team. 2022. No language left behind: Scaling human-centered machine translation. *Preprint*, arXiv:2207.04672.

OpenAI. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural 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 244–258, Online. Association for Computational Linguistics.

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, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. 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, Copenhagen, Denmark. Association for Computational Linguistics.

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.

Zhi Qu, Chenchen Ding, and Taro Watanabe. 2024a. Languages transferred within the encoder: On representation transfer in zero-shot multilingual translation. *Preprint*, arXiv:2406.08092.

Zhi Qu, Yiran Wang, Chenchen Ding, Hideki Tanaka, Masao Utiyama, and Taro Watanabe. 2024b. Improving language transfer capability of decoder-only architecture in multilingual neural machine translation. *Preprint*, arXiv:2412.02101.

Zhi Qu and Taro Watanabe. 2022. Adapting to non-centered languages for zero-shot multilingual translation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5251–5265, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, and 1 others. 2018. Improving language understanding by generative pre-training.

Ricardo Rei, José G. C. de Souza, Duarte Alves, Chrysoula Zerva, Ana C Farinha, Taisiya Glushkova, Alon Lavie, Luisa Coheur, and André F. T. Martins. 2022. COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

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.

Annette Rios, Mathias Müller, and Rico Sennrich. 2020. Subword segmentation and a single bridge language affect zero-shot neural machine translation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 528–537, Online. Association for Computational Linguistics.

David Stap, Vlad Niculae, and Christof Monz. 2023. Viewing knowledge transfer in multilingual machine translation through a representational lens. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14973–14987, Singapore. Association for Computational Linguistics.

Zengkui Sun, Yijin Liu, Fandong Meng, Jinan Xu, Yufeng Chen, and Jie Zhou. 2024. LCS: A language converter strategy for zero-shot neural machine translation. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 9201–9214, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Shaomu Tan and Christof Monz. 2023. Towards a better understanding of variations in zero-shot neural machine translation performance. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13553–13568, Singapore. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023. Aligning large language models with human: A survey. *Preprint*, arXiv:2307.12966.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Liwei Wu, Shanbo Cheng, Mingxuan Wang, and Lei Li. 2021. Language tags matter for zero-shot neural machine translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3001–3007, Online. Association for Computational Linguistics.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. 2020. On layer normalization in the transformer architecture. *Preprint*, arXiv:2002.04745.

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

Wen Yang, Chong Li, Jiajun Zhang, and Chengqing Zong. 2023. Bigtranslate: Augmenting large language models with multilingual translation capability over 100 languages. *Preprint*, arXiv:2305.18098.

Biao Zhang, Ankur Bapna, Rico Sennrich, and Orhan Firat. 2021. Share or not? learning to schedule language-specific capacity for multilingual translation. In *International Conference on Learning Representations*.

Biao Zhang, Behrooz Ghorbani, Ankur Bapna, Yong Cheng, Xavier Garcia, Jonathan Shen, and Orhan Firat. 2022. Examining scaling and transfer of language model architectures for machine translation. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 26176–26192. PMLR.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

Wenhao Zhu, Hongyi Liu, Qingxiu Dong, Jingjing Xu, Shujian Huang, Lingpeng Kong, Jiajun Chen, and Lei Li. 2023. Multilingual machine translation with large language models: Empirical results and analysis. *Preprint*, arXiv:2304.04675.

A Description of EC-40

EC-40 is an English-centric dataset introduced by Tan and Monz (2023). In addition to English, it includes 40 languages spanning five language families, with each family containing eight languages. These languages are categorized into four tiers based on data availability: High, Medium, Low, and Extra Low. Each non-English language is paired with English, resulting in 80 supervised translation directions used for training and 1,560 zero-shot translation directions. Details of this dataset are summarized in Table 12.

B Description of Pre-training Dataset

Our pre-training dataset comprises 24 languages, as detailed in Table 6. As described in Section 5.1, our data collection strategy results in 9.3 billion translation instances across 194 translation directions. The data distribution is visualized at the family level in Figure 8a and at the language level in Figure 8b. Additionally, Figure 8b highlights which translation directions are supervised and which are zero-shot. Notably, translation directions involving `ms` are also indicated in Figure 8b.

C Training Details of EC-40

Training configurations We employ Fairseq (Ott et al., 2019), an open-source toolkit, to implement our models with methods mentioned in Section 4.2. First, we directly reuse the vocabulary and binary training data provided by Chen et al. (2023)[16]. Note that we include only supervised translation directions in validation. We train on 8 Tesla V100 GPUs, setting *memory-efficient-fp16* in Fairseq, with a maximum input of 2048 source tokens per GPU and a gradient accumulation of 16 steps. Both input and output token lengths are limited to 256, and we share the embedding layer between the encoder and decoder. We use a seed of 1234, a learning rate of 0.0005 with the inverse square root schedule and a warmup of 4000 steps, the Adam optimizer (Kingma and Ba, 2017), dropout of 0.1, attention dropout of 0.1, a label smoothing rate of 0.1, no weight decay, and the temperature sampling with $T = 5$ (Arivazhagan et al., 2019). Finally, we train for 200,000 steps, averaging the last 5 checkpoints, saving by epoch.

Configurations of Related Methods Generally, both our method and these related methods share the same hyper-parameters as the vanilla models. However, there are some method-specific configurations we have to notice. (1) For LAVS (Chen et al., 2023), we directly reuse their code[17] and add around 12k language-specific tokens into the shared vocabulary, resulting in 12.8M additional parameters in modeling; (2) For CL (Pan et al., 2021), we directly reuse their code[18] and set the contrastive learning temperature to 0.1, which is the optimal setting according to their reports; (3) For LCS (Sun et al., 2024), the model follows another translation instruction strategy of Fan et al. (2020) by adding a source language tag at the beginning of the source tokens and a target language tag at the beginning of the target tokens. we re-implement their code and, in the case of 12-layer models where the encoder has 6 layers, apply LCS biasing at the 5th encoder layer; For models with 12 encoder layers, we apply it at the 8th encoder layer; (4) For TDO (Qu et al., 2024b), we also reuse their code[19] and, based on their ablation study, set the number of layers for the first stage to 3 in 12-layer models and to 6 in 24-layer models to allow stronger zero-shot translation ability.

Family	ISO code	Flores code	Language	script
Germanic	en	eng_Latn	English	Latin
	de	deu_Latn	German	Latin
	nl	nld_Latn	Dutch	Latin
	sv	swe_Latn	Swedish	Latin
	da	dan_Latn	Danish	Latin
	af	afr_Latn	Afrikaans	Latin
Romance	fr	fra_Latn	French	Latin
	es	spa_Latn	Spanish	Latin
	it	ita_Latn	Italian	Latin
	pt	por_Latn	Portuguese	Latin
	ro	ron_Latn	Romanian	Latin
Slavic	ru	rus_Cyrl	Russian	Cyrillic
	cs	ces_Latn	Czech	Latin
	pl	pol_Latn	Polish	Latin
	bg	bul_Cyrl	Bulgarian	Cyrillic
	uk	ukr_Cyrl	Ukrainian	Cyrillic
Malayo-Polynesian	id	ind_Latn	Indonesian	Latin
	ms	zsm_Latn	Malay	Latin
	jv	jav_Latn	Javanese	Latin
	tl	fil_Latn	Filipino	Latin
Asian*	ko	kor_Hang	Korean	Hangul
	vi	vie_Latn	Vietnamese	Latin
	ja	jpn_Jpan	Japanese	Kanji; Kana
	zh	cmn_Hans	Chinese	Chinese

Table 6: Details of the dataset in our pre-training. The decoration * on Asian means a group instead of a language family. We not only list the ISO 630-1 code for each language but also list the Flores code to help search corresponding resources from Flores+.

[16] https://github.com/Smu-Tan/ZS-NMT-Variations

[17] https://github.com/PKUnlp-icler/Off-Target-MNMT

[18] https://github.com/PANXiao1994/mRASP2

[19] https://github.com/zhiqu22/PhasedDecoder

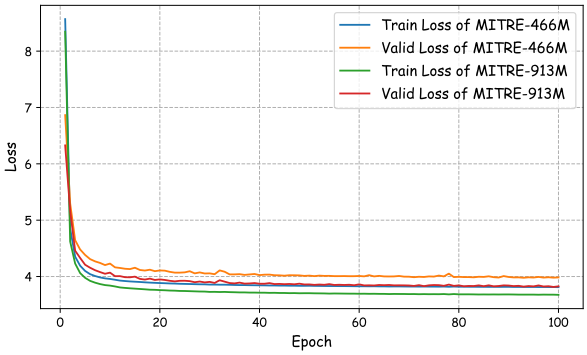


Figure 6: The training and validation loss in pre-training MITRE. We report the first 100 epochs, each with approximately 2262 steps.

D Training Details of MITRE

We employ Fairseq to implement MITRE mentioned in Section 5.2, and two versions of MITRE have the different configurations in modeling and have the same configuration in training. Specifically, MITRE-466M is configured with an embedding size of 1,024, an inner size of 4,096, 16 attention heads, and 24 layers. MITRE-913M, a larger model with expanded width and depth, has an embedding size of 1,280, an inner size of 5,120, 20 heads, and 36 layers. In training, we first train a shared SentencePiece vocabulary (Kudo and Richardson, 2018) with a size of 160,000 by 150 million sentences randomly sampled from the training set. We include only supervised translation directions in validation. Then, we train on 80 Tesla V100 GPUs, setting *memory-efficient-fp16* in Fairseq, with a maximum input of 1408 source tokens per GPU and a gradient accumulation of 10 steps. In practice, this setup results in each batch containing approximately 0.91 million source tokens. Given the large batch size, we set the learning rate of 0.002 with the inverse square root schedule and the warmup of 8000 steps. We also use a seed of 42, the Adam optimizer (Kingma and Ba, 2017), dropout of 0.1, attention dropout of 0.1, a label smoothing rate of 0.1, no weight decay, and the temperature sampling with $T = 1$ (Arivazhagan et al., 2019). We train for 300,000 steps and save a checkpoint per 10,000 steps. Finally, we average the last 5 checkpoints. Figure 6 shows the variations of training and validation loss. We can observe that the trends of MITRE-466M and MITRE-913M are highly consistent.

E Training Details of Fine-tuning

Selecting Directions We use *random.sample* in Python to randomly select translation directions for fine-tuning, setting the seed to 0. We define three scenarios, including 5, 25, and 100 translation directions. It is important to note that *random.sample* causes the 5 and 25 directions to be subsets of the 100 directions. Specifically, the first 5 and first 25 directions in the 100-direction set correspond to the other two scenarios. The 100 directions are: jv→sv, ms→id, de→tl, ru→pl, ko→jv, zh→bg, ms→en, pl→ru, zh→af, uk→ko, pt→jv, ko→ro, fr→da, cs→pl, fr→af, da→fr, ru→sv, fr→pl, pl→tl, da→ro, sv→es, bg→jv, zh→en, da→cs, uk→ms, tl→es, bg→de, pt→nl, vi→bg, tl→id, ru→bg, nl→ms, en→uk, da→sv, jv→ms, en→nl, zh→vi, bg→ja, ro→ja, bg→ru, nl→tl, vi→es, ja→pt, cs→uk, da→ko, af→it, jv→zh, zh→cs, sv→da, ko→pt, cs→nl, pt→vi, nl→en, vi→ja, es→nl, tl→ru, ru→es, ja→jv, ro→zh, nl→ro, fr→jv, cs→fr, fr→cs, uk→jv, ko→bg, cs→da, es→ro, ms→sv, ja→cs, cs→en, da→pl, jv→tl, pl→pt, zh→sv, pl→de, fr→ro, pt→zh, zh→id, pl→fr, ko→ru, it→bg, es→de, cs→tl, af→pt, fr→ru, da→nl, da→af, ms→fr, ko→cs, en→jv, pl→uk, bg→uk, af→tl, ro→bg, de→pl, de→vi, uk→nl, id→ja, nl→zh, zh→pl

Fine-tuning Configurations All fine-tuning experiments use the same settings. We conduct experiments on 8 Tesla V100 GPUs, with a maximum of 1024 tokens per GPU and a gradient accumulation of 2 steps. Based on the pre-trained model, we set the learning rate to 0.0001 with a warmup step of 1 (for launching the inverse square root schedule), and train for 10 epochs. Finally, we use the last epoch for testing.

LoRA Configurations We adopt the setting of Hu et al. (2022) to implement LoRA components for pre-trained models. Specifically, LoRA is only implemented for Query and Value in the attention mechanism with a rank of 8. As a result, the learnable parameters of NLLB series are 1.18M, 2.36M, and 4.72M, respectively, and the learnable parameters of MITRE series are 0.78M and 1.47M, respectively.

F Prompts for GPT

Our prompts for GPT series follow: Translating the following sentence from [SRC] to [TGT]: [INPUT]. Here, [SRC] and [TGT] are the source and target

language names following Table 6, and [INPUT] is the source sentence. We find that GPT occasionally repeats [INPUT] in the output. Once it happens, we manually remove the [INPUT] before evaluation.

G Details of Evaluation Metrics

In evaluating the performance of models trained on EC-40, some languages lack support from COMET (*Unbabel/wmt22-comet-da*) and the off-target ratio (*fast-langdetect*). Notably, *fast-langdetect* operates by word recognition, so we also exclude certain supported languages that exhibit low recognition success rates. We list the supported languages in this section.

Languages in COMET: en, bg, so, ca, da, be, bs, es, uk, am, hi, ro, no, de, cs, pt, nl, mr, is, ne, ur, ha, sv, gu, ar, fr, ru, it, pl, sr, sd, he, af, kn, bn.

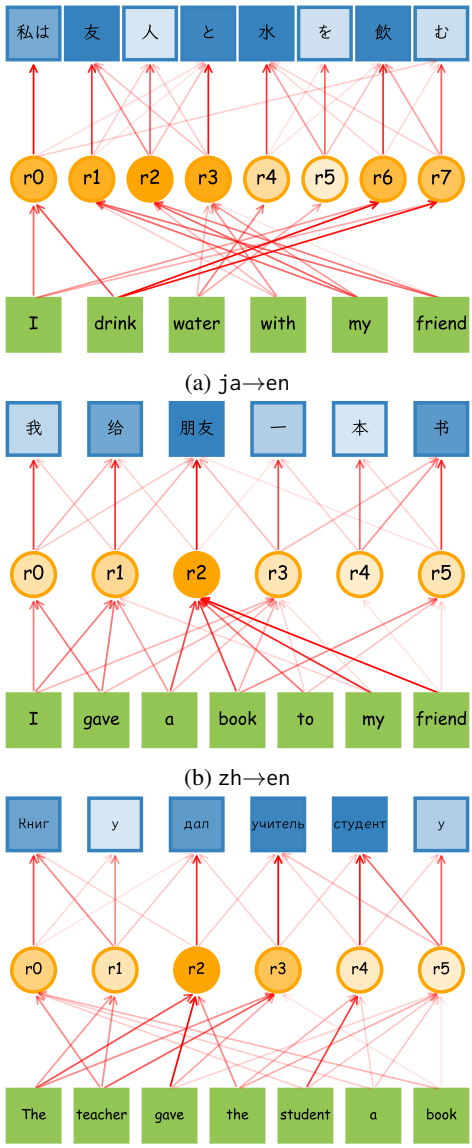
Languages in Off-target Ratio: en, bg, da, es, uk, hi, ro, de, cs, pt, nl, mr, ur, sv, gu, ar, fr, ru, it, pl, he, kn, bn, be, mt, am, is, sd.

H Supplementary Results of EC-40

In Section 4.3 and Table 1, we report spBLEU scores and off-target ratio. In this appendix, we report chrF++ and COMET scores in Tables 8 and 9, respectively. Overall, four metrics show consistent trends across this benchmark.

I Supplementary Results of MITRE

In Section 5.3 and Table 3, we report spBLEU scores and do not report the off-target ratio, because the values are near zero across those large-scale pre-trained models. In this appendix, we report chrF++ and COMET scores in Tables 10 and 11, respectively. However, when comparing MITRE and commercial LLMs, COMET reveals a different trend: MITRE-913M underperforms GPT-4o mini, despite similar trends in spBLEU and chrF++ scores, which show MITRE-913M as superior to GPT-4o mini. This suggests that while MITRE generates more accurate sequences relative to the test set, GPT-4o mini produces more fluent and natural output. This is expected, as commercial models are aligned with human-like styles (Wang et al., 2023), whereas MITRE follows the training data's style. Further supporting this, GPT-4o mini shows a significant improvement over GPT-3.5 turbo (Table 11). Therefore, based on the results across all three metrics, our claim is not that MITRE-913M outperforms GPT-4o mini, but rather that MITRE-913M outperforms NLLB 3.3B and can compete with GPT-4o mini.



(a) ja→en

(b) zh→en

(c) ru→en

Figure 7: Three cases of attention analysis on MITRE-466M. Details of this illustration, e.g. colors, classes, and arrows, follow that of Figure 5.

J Supporting Experiments for Comparing Enc-dec and Dec-only

In Section 3.2, we explain our reason for implementing registering in Dec-only. Specifically, Dec-only offers better parameter efficiency than Enc-dec, where the encoder learns the representation of input tokens while the decoder learns the generated tokens (as illustrated in Figure 1, where encoded source tokens are shown in gray). In contrast, Dec-only utilizes all parameters for both encoding and

Methods	#layer	spBLEU	chrF++	COMET	Off-target(%)
Enc-dec	12	6.99	19.68	53.72	48.4
+registering		10.55	27.29	58.07	7.5
Dec-only		8.34	22.34	55.71	22.21
+registering		**11.92**	**29.02**	**61.19**	**4.65**
Enc-dec	24	9.69	23.95	58.08	26.69
+registering		11.13	27.91	58.67	3.91
Dec-only		10.82	26.04	59.95	19.01
+registering		**13.12**	**30.51**	**63.54**	**3.65**

Table 7: Results on EC-40. For convenience, the score in this table is averaged from all 1,640 translation directions. Enc-dec and Dec-only indicate the vanilla models without registering. The best score is in bold.

generation. Based on this, we employ Dec-only as the backbone of our implementation.

To further support our statement, we provide supporting experiments on EC40, following the experimental setup described in Section 4.2. As shown in Table 7, registering also significantly improves the performance of Enc-dec. However, the gains are more pronounced in Dec-only, suggesting that registering is particularly well-suited for this architecture.

Additionally, we propose an insight beyond the scope of this work. In Section 3.2, we mention a potential cause of the off-target problem: the dilution of translation instruction attention by other tokens (Gu et al., 2019; Qu et al., 2024b). This theory may explain why Dec-only tends to underperform Enc-dec in MNMT (Gao et al., 2022; Zhang et al., 2022), as the effectiveness of $l_{\boldsymbol{y}}$ is further diluted by both source and target tokens in the attention mechanism. The results in Table 7 further support this explanation, as registering achieves a much larger performance gain in Dec-only than in Enc-dec.

K Supplementary Analysis of Attention

To further support our analysis in Section 6.2, we examine additional cases in MITRE-466M where the source and target sentences exhibit significant structural differences. In all cases, the attention relationship between registers and source tokens remains consistent, i.e., one-to-one attention weights being the most prominent. Next, we observe the following patterns: (1) As shown in Figure 7a, in Japanese, the attention for "drink" points to r_6, while "friend" points to r_1 and r_2. (2) As shown in Figure 7b, "friend" points to r_2, and "book" points to r_5. (3) As shown in Figure 7c, "book" points to r_0, and "student" points to r_4. Given that the attention weights between registers and target tokens highlight the structural differences between source and target sentences, we can state again that registers mirror the corresponding source tokens.

L Supplementary Analysis of Representation

We present Figure 9 to supplement the analysis in Section 6.2 on representation distributions, where Figure 4 focuses specifically on the representation state in the 24th layer. Our observations are as follows: (1) In the embedding layer and the 1st layer output (Figures 9a and 9b), source and target token representations are loosely distributed, while registers form three compact clusters based on language. This is because registers lack semantic content and are distinguished only by positional encoding. (2) Starting from the 6th layer (Figure 9c), source tokens begin to become distinguishable by language, and registers start to shift within the representation space toward the source tokens. By the 12th layer (Figure 9d), registers and source tokens are entirely separated in the representation space. (3) By the 18th layer (Figure 9e), target tokens become clearly separated in the representation space, registers' distribution becomes more diffuse, and the distribution of source tokens becomes more concentrated. These trends culminate in the state observed in the 24th layer (Figure 9f), as described in Section 6.2. These findings suggest two key phenomena: (1) registers progressively reinforce the semantic information they carry as they propagate through the layers; and (2) the representations of target tokens reflect their predicted state only in the higher layers.

			High		Med		Low		Extra Low				
	#params	Method	→	←	→	←	→	←	→	←	sup.	zero.	avg.
Enc-dec	242M	vanilla	23.27	25.61	20.43	24.61	16.39	14.25	17.73	13.34	49.27	18.16	19.68
		+CL	31.64	30.87	28.65	31.27	21.38	21.66	21.46	19.33	49.20	24.86	26.04
		+LCS	25.34	31.05	24.13	30.90	24.81	19.84	24.77	17.45	**49.34**	23.70	24.95
	259M	+LAVS	24.99	26.66	22.33	25.77	17.55	15.54	17.77	14.64	49.29	19.38	20.88
Dec-only	217M	vanilla	27.26	27.10	24.42	27.16	18.30	18.01	18.46	16.17	48.44	21.00	22.34
		+TDO	30.25	29.87	25.87	29.61	20.61	19.75	20.50	18.00	48.92	23.32	24.57
		+Ours	**33.74**	**33.55**	**31.77**	**32.47**	**24.87**	**24.50**	**24.82**	**24.68**	48.87	**28.00**	**29.02**
Enc-dec	418M	vanilla	27.91	31.84	25.40	31.48	20.57	16.02	20.79	15.33	50.11	22.60	23.95
		+CL	33.02	32.52	29.85	32.84	20.99	22.05	21.84	18.30	**50.41**	25.51	26.73
		+LCS	24.67	33.81	23.88	32.99	26.30	18.10	25.71	16.87	50.16	24.26	25.57
	430M	+LAVS	29.69	34.49	27.68	33.90	22.66	18.62	23.75	15.77	49.89	25.02	26.18
Dec-only	368M	vanilla	31.01	31.96	28.04	32.20	22.05	20.13	22.02	18.84	49.76	24.82	26.04
		+TDO	32.12	32.22	28.43	32.24	21.84	20.85	22.25	19.32	50.04	25.23	26.44
		+Ours	**35.24**	**35.00**	**33.31**	**34.56**	**26.41**	**26.01**	**26.22**	**25.61**	49.69	**29.53**	**30.51**

Table 8: Averaged chrF++ scores of results on EC-40. All notations and abbreviations follow Table 1.

			High		Med		Low		Extra Low				
	#params	Method	→	←	→	←	→	←	→	←	sup.	zero.	avg.
Enc-dec	242M	vanilla	50.21	49.61	42.07	44.93	28.62	28.15	32.94	31.17	77.26	52.29	53.72
		+CL	55.33	52.42	46.33	47.60	30.00	31.42	34.17	34.40	77.33	56.75	57.93
		+LCS	50.27	52.22	43.19	47.73	31.88	30.19	37.09	32.50	**77.52**	55.44	56.71
	259M	+LAVS	52.89	51.57	44.30	46.13	28.81	28.71	32.97	32.16	77.46	54.64	56.08
Dec-only	217M	vanilla	52.66	50.88	43.80	45.88	29.19	30.13	33.88	32.64	77.10	54.41	55.71
		+TDO	54.39	52.48	45.22	47.40	30.09	30.76	34.53	33.59	77.13	56.16	57.36
		+Ours	**56.94**	**54.87**	**48.74**	**49.59**	**32.61**	**33.83**	**37.10**	**37.11**	77.12	**60.23**	**61.19**
Enc-dec	418M	vanilla	54.06	54.76	45.71	49.32	31.11	29.68	35.37	32.49	78.48	56.84	58.08
		+CL	58.06	55.20	48.40	49.96	30.79	32.53	35.50	35.06	**78.90**	59.26	60.38
		+LCS	51.52	55.63	44.84	50.74	33.84	30.80	**38.91**	33.94	78.62	57.61	58.81
	430M	+LAVS	54.18	56.30	46.24	50.73	33.02	31.60	37.64	32.46	76.29	57.90	58.95
Dec-only	368M	vanilla	56.46	55.29	47.26	50.07	31.66	31.85	36.27	34.44	78.37	58.84	59.95
		+TDO	57.49	55.44	48.34	50.48	31.77	32.57	36.61	35.72	78.48	59.77	60.84
		+Ours	**58.98**	**56.92**	**50.66**	**51.82**	**33.91**	**35.37**	38.60	**38.04**	78.12	**62.66**	**63.54**

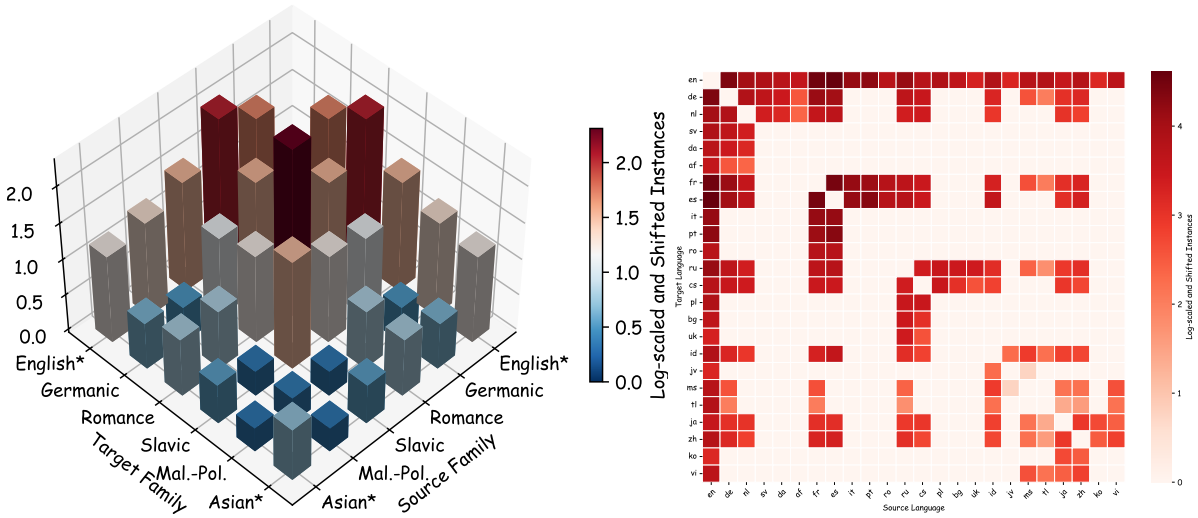
Table 9: Averaged COMET scores of results on EC-40. All notations and abbreviations follow Table 1.

		English		Germanic		Romance		Slavic		Mal.-Polyn.		Asian		
Model		→	←	→	←	→	←	→	←	→	←	→	←	avg.
M2M	483M	50.43	54.36	44.84	46.24	43.96	48.26	43.36	43.06	37.54	41.77	39.83	27.85	42.53
	615M	49.97	54.74	46.77	48.62	45.34	49.83	44.70	44.42	40.17	43.59	41.04	28.84	44.12
	1.2B	54.80	54.44	49.02	47.06	47.49	48.04	45.87	43.35	32.78	40.68	30.90	28.01	42.56
NLLB	615M	55.54	61.73	48.48	50.39	47.38	51.54	46.38	45.34	45.98	50.96	42.56	29.73	46.70
	1.3B	56.82	63.35	50.07	52.21	48.80	53.19	47.98	47.46	47.92	52.49	44.70	30.98	48.40
	3.3B	57.88	**64.27**	50.95	53.16	49.63	53.92	48.91	48.76	49.06	**53.28**	45.71	31.97	49.35
GPT	3.5 turbo	55.30	61.60	49.39	52.16	48.31	53.34	47.54	46.35	45.64	48.05	43.46	31.20	47.41
	4o mini	58.03	62.85	51.26	52.90	49.33	54.33	49.12	48.62	**49.39**	52.25	45.86	**34.11**	49.48
MITRE	466M	58.11	62.77	51.40	53.41	50.06	54.46	49.18	48.62	47.83	52.04	45.10	32.41	49.29
	913M	**58.84**	64.01	**52.40**	**54.59**	**51.03**	**55.36**	**50.32**	**49.84**	48.97	52.88	**46.65**	33.88	**50.42**

Table 10: Averaged chrF++ scores of results for comparing MITRE and baselines. All notations and abbreviations follow Table 3.

		English		Germanic		Romance		Slavic		Mal.-Polyn.		Asian		
Model		→	←	→	←	→	←	→	←	→	←	→	←	avg.
M2M	483M	81.63	81.40	78.90	77.03	80.48	78.49	79.85	80.31	68.34	72.75	78.67	78.57	77.74
	615M	81.16	82.15	81.42	79.98	82.00	80.50	81.29	82.43	72.56	74.62	80.02	79.96	79.79
	1.2B	85.93	85.17	84.15	82.87	84.46	83.12	83.87	85.41	75.91	77.83	82.95	82.58	82.66
NLLB	615M	86.61	86.76	84.06	82.77	84.50	83.05	83.41	84.39	81.26	83.51	82.12	82.02	83.33
	1.3B	87.76	87.63	85.72	84.76	85.93	84.76	85.07	86.82	83.57	84.93	84.36	83.52	85.13
	3.3B	88.22	88.09	86.49	85.58	86.58	85.45	85.84	87.96	84.63	85.45	85.42	84.56	85.96
GPT	3.5 turbo	87.67	88.02	86.26	85.80	86.62	85.96	85.91	87.50	83.09	82.09	85.45	85.77	85.66
	4o mini	**89.59**	**88.64**	**87.50**	**86.38**	**87.58**	**86.57**	**87.08**	**88.90**	**85.89**	**86.03**	**86.99**	**87.47**	**87.16**
MITRE	466M	87.87	87.29	85.99	84.98	86.49	85.14	85.58	87.19	82.24	83.41	84.38	84.29	85.19
	913M	88.11	87.81	86.54	85.61	86.96	85.70	86.16	88.03	83.15	83.80	85.52	85.35	85.88

Table 11: Averaged COMET scores of results for comparing MITRE and baselines. All notations and abbreviations follow Table 3. Given the different trend compared to Tables 3 and 10, we not only mention it in Section 5.3, but also provide an additional discussion in Appendix I.



(a) Family Level

(b) Language Level

Figure 8: Data distribution of our pre-training dataset. Figure 8a shows data size distribution at the family level, while Figure 8b displays data size at the language level. In 8a, non-zero values are scaled by log10 and adjusted by subtracting 7 for clearer visualization. In 8b, non-zero values are also scaled by log10 and shifted by subtracting the minimum value to enhance illustration clarity.

	Germanic			Romance			Slavic			Indo-Aryan			Afro-Asiatic		
	code	Language	Script	code	Language	Script	code	Language	Script	code	Language	Script	code	Language	Script
High	de	German	Latin	fr	French	Latin	ru	Russian	Cyrillic	hi	Hindi	Devanagari	ar	Arabic	Arabic
(5 million)	nl	Dutch	Latin	es	Spanish	Latin	cs	Czech	Latin	bn	Bengali	Bengali	he	Hebrew	Hebrew
Med	sv	Swedish	Latin	it	Italian	Latin	pl	Polish	Latin	kn	Kannada	Devanagari	mt	Maltese	Latin
(1 million)	da	Danish	Latin	pt	Portuguese	Latin	bg	Bulgarian	Cyrillic	mr	Marathi	Devanagari	ha	Hausa*	Latin
Low	af	Afrikaans	Latin	ro	Romanian	Latin	uk	Ukrainian	Cyrillic	sd	Sindhi	Arabic	ti	Tigrinya	Ethiopic
(100 thousand)	lb	Luxembourgish	Latin	oc	Occitan	Latin	sr	Serbian	Latin	gu	Gujarati	Devanagari	am	Amharic	Ethiopic
Extra Low	no	Norwegian	Latin	ast	Asturian	Latin	be	Belarusian	Cyrillic	ne	Nepali	Devanagari	kab	Kabyle*	Latin
(50 thousand)	ic	Icelandic	Latin	ca	Catalan	Latin	bs	Bosnian	Latin	ur	Urdu	Arabic	so	Somali	Latin

Table 12: Details of non-English languages in EC-40. This table is duplicated from Tan and Monz (2023). Numbers in the table represent the number of sentences paired to the English. Two exceptions are Hausa and Kabyle, where their data sizes are 334,000 and 18,448, respectively.

(a) Embedding Layer

(b) 1st Layer

(c) 6th Layer

(d) 12th Layer

(e) 18th Layer

(f) 24th Layer

Figure 9: 2D distributions of token-level representations extracted from the different layers of a model trained on EC-40. This illustration complements Figure 4.