TRANS-ZERO: Self-Play Incentivizes Large Language Models for Multilingual Translation Without Parallel Data

Wei Zou[♠†] **Sen Yang**[♠†] **Yu Bao**[♡] **Shujian Huang**[♠*] **Jiajun Chen**[♠] **Shanbo Cheng**[♡*]
[♠]National Key Laboratory for Novel Software Technology, Nanjing University
[♡]ByteDance Research
{zouw,yangsen}@smail.nju.edu.cn, {huangsj, chenjj}@nju.edu.cn
{baoyu.001,chengshanbo}@bytedance.com

Abstract

The rise of Large Language Models (LLMs) has reshaped machine translation (MT), but multilingual MT still relies heavily on parallel data for supervised fine-tuning (SFT), facing challenges like data scarcity for low-resource languages and catastrophic forgetting. To address these issues, we propose TRANS-ZERO, a self-play framework that leverages only monolingual data and the intrinsic multilingual knowledge of LLM. TRANS-ZERO combines a novel Monte-Carlo Tree Search, G-MCTS, with preference optimization, achieving strong translation performance that rivals supervised methods. Experiments demonstrate that this approach not only matches the performance of models trained on large-scale parallel data but also excels in non-English translation directions. Further analysis reveals that G-MCTS itself significantly enhances translation quality by exploring semantically consistent candidates through iterative translations, providing a robust foundation for the framework's success. Codes are available at https://github.com/NJUNLP/trans0

1 Introduction

The advent of Large Language Models (LLMs) witnesses a fundamental shift in machine translation (MT) paradigms from the supervised end-to-end training (Vaswani et al., 2017) to the sophisticated generation of fine-tuned language models (Achiam et al., 2023).

Unlike various downstream tasks that gain impressive proficiency through lightweight instruction tuning, multilingual translations between languages still necessitate sufficient fine-tuning with parallel data for specific translation directions. Besides external human annotations, researchers like Xu et al. (2024c); Li et al. (2024) obtain translation annotations or preferences from external LLM, as long as they prove multilingual capability. Either way, it faces a notable deficit in data scarcity for less popular languages. Moreover, issues arise as the multilingual translation fine-tuning scales up. The reliance on one-on-one MLE supervision has been criticized for potential biases that clash with natural language's inherent multilingualism (Zhu et al., 2024a) and poses a risk of catastrophic forgetting. Furthermore, exceeding multilingual annotations inversely dilutes the pre-trained knowledge in supported languages, thus degrading overall cross-lingual performance (Xu et al., 2024a; Zhu et al., 2024b). Xu et al. (2024b) proposes a mixture-of-expert with hand-crafted route across language modules. However, the route and distributed overheads increase exponentially as the number of translation directions involved increases.

Whereas traditional fine-tuning scaling approaches a plateau, leveraging LLMs' inherent knowledge rather than external supervision for self-improvement is trending (Chen et al., 2024; Kumar et al., 2024). However, adapting this approach to MT introduces two technical challenges. First, systematic *cross-lingual exploration* requires navigating complex semantic spaces beyond simple prompt engineering. Traditional LLM planning involves delicate prompt-based reasoning, even fine-tuning, which is generally unavailable for most scenarios. Second, *multilingual quality assessment* must overcome the limitations of data-dependent quality estimation (QE) metrics and reward model training complexities.

In this work, we introduce TRANS-ZERO. This innovative self-play framework enables LLMs to bootstrap their multilingualism by strategically exploring their semantic space, achieving self-improvement for multilingual translation given only monolingual data. We start by defining a Multilingual Translation Process (MTP) that gen-

[*]Corresponding author
[†]Work done during internship at ByteDance Research

Findings of the Association for Computational Linguistics: ACL 2025, pages 12337–12347
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

erates translations by interweaving the languages supported by LLM, so the inference is scaled up to explore more potential translations. First, we implement the Genetic Monte-Carlo tree search (G-MCTS) upon MTP, exploring potential translations. Second, we harness the search to assess translation preferences based on intrinsic multilingual consistency. Experiments verify that TRANS-ZERO improves the lesser translations through iterative G-MCTS and preference optimizations given only monolingual data. We summarize our contributions as follows:

- First self-play framework extending to multilingual MT training with monolingual data.
- Novel integration of MCTS that explores improved translation for preference.
- An intrinsic translation preference without additional QE modules enables iterative self-improvement.

2 Preliminary

2.1 Machine Translation via LLM

Traditional machine translation depends on large parallel supervision in specific language pairs, which is limited by insufficient annotation in less popular translation directions. The emergence of multilingual pre-trained language models has revolutionized the field, enabling impressive performance across various downstream tasks with minimal supervision (Wei et al., 2022a). State-of-the-art (SOTA) LLM-based translation systems now achieve competitive results with significantly fewer annotations by leveraging supervised fine-tuning guided by sophisticated instructions (see Appendix A for details). Notably, Zhu et al. (2024b) demonstrated that LLMs exhibit remarkable zero-shot and few-shot machine translation capabilities, even without explicit instruction formatting or exemplars. This highlights the intrinsic potential of LLMs for self-improvement beyond finely calibrated supervision, opening new avenues for resource-efficient translation paradigms.

2.2 Monte-Carlo Tree Search

Monte-Carlo Tree Search (MCTS, Browne et al., 2012; Świechowski et al., 2023) is a heuristic search algorithm proficient for complex decision processes such as the game of Go (Silver et al., 2016). MCTS proceeds through four steps: selection, expansion, simulation, and backpropagation.

- **Selection.** MCTS descends the tree from its root by selecting the top amongst child nodes by their upper confidence bounds (UCB):

$$\mathrm{UCB}(\alpha) = \nu(\alpha) + 2\sqrt{\frac{\log N(A)}{1+N(\alpha)}}, \quad (1)$$

where node α is a child of node A, and $N(*)$ is the node's visit count, with $\nu(*)$ as its current utility. The utility of α is the cumulated rewards $Q(\alpha)$ averaged by its visit count:

$$\nu(\alpha) = \frac{Q(\alpha)}{N(\alpha)} \quad (2)$$

The UCB balances the exploration and exploitation in the heuristic search.

- **Expansion.** Upon reaching a selected node, MCTS expands the tree by adding a new child node representing a possible move from the current state.
- **Simulation.** From the newly expanded node, a random simulation is performed until either the decision process concludes or reaches a maximum step limit, estimating the reward r for this decision path.
- **Backpropagation.** The obtained reward is propagated backward, updating the utility values of all nodes along the path from the expanded node to the root. This update refines the UCB values, progressively enhancing the efficiency of subsequent search iterations.

2.3 Self-Optimization in LLM

Typically, optimizations for LLM rely on external rewards (e.g., critic and revise modules) for tuning (Huang et al., 2023; Tian et al., 2024; Zhang et al., 2024). Recent work explores self-optimization using LLMs' internal knowledge, leveraging performance gaps across test scenarios. For example, in multilingual tasks, higher-performing languages can optimize lesser ones (Geng et al., 2024). She et al. (2024) uses a strong language as a pivot to map and optimize weaker languages. Self-play preference optimization (Chen et al., 2024, SPPO) adopts the gaming theory where the post-update models shall prevail by a win rate $\mathbb{P}(\cdot)$ to optimize the preference. The SPPO is apt for pairwise preference $(y_w \succ y_l, x)$

in the following symmetric loss:

$$\mathcal{L}_{\text{SPPO}}(x, y_w, y_l, \theta, \pi_{\text{ref}}) \tag{3}$$
$$= [\log \frac{\pi_\theta(y_w|x)}{\pi_{\text{pre}}(y_w|x)} - \eta(\mathbb{P}(y_w \succ y_l|x) - \frac{1}{2})]^2$$
$$- [\log \frac{\pi_\theta(y_l|x)}{\pi_{\text{pre}}(y_l|x)} - \eta(\mathbb{P}(y_l \succ y_w|x) - \frac{1}{2})]^2,$$

where θ is the LLM's parameters, π_{pre} is the LLM before update and η is a hyperparameter. SPPO suits the human preferences' non-transitive, unstable nature, thus enabling sustainable self-play optimization.

3 TRANS-ZERO

In this section, we present TRANS-ZERO. First, we introduce the Multilingual Translation Process (MTP), a novel framework orchestrating multi-step translation across multiple languages (§3.1). Second, we implement Genetic Monte Carlo Tree Search (G-MCTS) upon MTP to explore promising translations (§3.2), which derive preference from cross-lingual semantic consistency in the search purely through translation prompts, eliminating the need for explicit reasoning training or reward learning. Finally, we utilize the search results for further preference optimization (§3.3), enabling the unsupervised MT training given only monolingual data.

3.1 Multilingual Translation Process

We define the Multilingual Translation Process (MTP) as an iterative translation involving at least two languages, denoted as $\{L_i\}_{|\{L_i\}|>1}$, where each translation step maps the sentence from one language to another distinct language within the set. An MTP trajectory of length T starts from a source language l:

$$l_T = f(l|l_1, l_2, ..., l_{T-1}),$$

where the l_i is a sentence in one language in $\{L_i\}$, and $f(\cdot)$ is the translation function. MTP iteratively scales up the translation across languages, enabling similar cross-lingual preferences in work by Geng et al. (2024); She et al. (2024).

Notably, an optimized translation ensures that *the semantics are maintained throughout the multilingual translations*. Such consistency preference intuitively guides the search toward better translation candidates, which can also contribute to preference optimization. E.g., translation optimized via round-trip translation promotes the bilingual semantic consistency given an MTP $x \to y \to x'$.

3.2 Genetic Monte-Carlo Tree Search

As shown in Figure 1, we conduct a genetic Monte-Carlo tree search based on the defined MTP. The search employs genetic expansion coupled with semantic consistency simulation. Each tree node corresponds to a translation candidate in the target language and is generated autoregressively by translations from existing nodes[1].

Initialization The input language is X with input text x, and the output language is Y. We initialize the search tree with x as the root and perform top-k sampling with a width of b to generate b translation candidates $\{y_i\}_b$ in language Y. Each candidate is assigned as a child node of the root. These nodes are quickly initialized by back-translation to language X, with the corresponding reconstruction recorded, denoted as x'. We define the consistency function $\mathrm{S}(x, x')$ over sentence pair (x, x') of **same** language based on a mutual evaluation metric $\mathrm{M}(x, x')$, e.g. BLEURT (Sellam et al., 2020). The consistency score is computed as:

$$\mathrm{S}(x, x') = \frac{\mathrm{M}(x, x') + \mathrm{M}(x', x)}{2}, \tag{4}$$

where x' is to compute fast-initiated reward $r(y_i) = \mathrm{S}(x, x')$, followed by backpropagations.

Genetic Expansion Given an initialized search tree, each expansion step of the MCTS selects the node with the highest UCB value to generate a new translation in the target language. However, a straightforward generation does not naturally lead to diverse exploration. Inspired by genetic algorithms (Sastry et al., 2005), we propose two strategies for expansion based on the status of the current maximum UCB node:

- **Merge.** We perform a merging when the current maximum UCB node differs from the maximum utility node: Merging is a few-shot translation given the current best translation (i.e., the maximum utility node) as demonstrations:

$$y_t = f(x \mid y_{\text{UCB}}, y_\nu)$$
$$y_{\text{UCB}} = \arg\max_{y<t}\{\text{UCB}(y)\}$$
$$y_\nu = \arg\max_{y<t}\{\nu(y)\}$$

Specifically, (x, y_ν) and (x, y_{UCB}), the pairs from both the maximum utility node and the

[1] The translation instructions during G-MCTS are sampled from Table 4 in Appendix A.

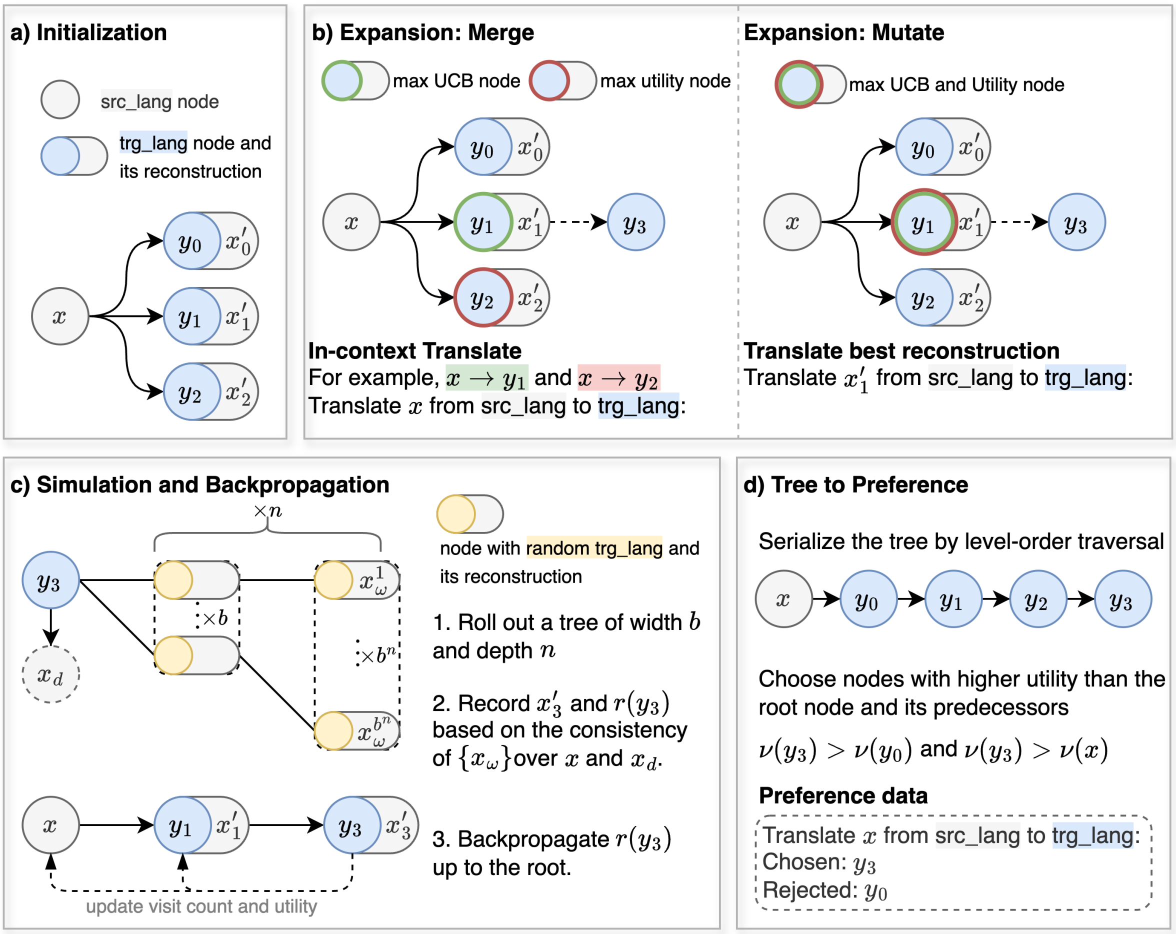
a) Initialization
src_lang node
trg_lang node and its reconstruction
x
y_0 x'_0
y_1 x'_1
y_2 x'_2
b) Expansion: Merge
max UCB node
max utility node
x
y_0 x'_0
y_1 x'_1
y_2 x'_2
y_3
In-context Translate
For example, $x \to y_1$ and $x \to y_2$
Translate x from src_lang to trg_lang:
Expansion: Mutate
max UCB and Utility node
x
y_0 x'_0
y_1 x'_1
y_2 x'_2
y_3
Translate best reconstruction
Translate x'_1 from src_lang to trg_lang:
c) Simulation and Backpropagation
$\times n$
y_3
x_d
$\times b$
$\times b^n$
x^1_ω
$x^{b^n}_\omega$
node with random trg_lang and its reconstruction
1. Roll out a tree of width b and depth n
2. Record x'_3 and $r(y_3)$ based on the consistency of $\{x_\omega\}$over x and x_d.
x
y_1 x'_1
y_3 x'_3
3. Backpropagate $r(y_3)$ up to the root.
update visit count and utility
d) Tree to Preference
Serialize the tree by level-order traversal
x y_0 y_1 y_2 y_3
Choose nodes with higher utility than the root node and its predecessors
$\nu(y_3) > \nu(y_0)$ and $\nu(y_3) > \nu(x)$
Preference data
Translate x from src_lang to trg_lang:
Chosen: y_3
Rejected: y_0

Figure 1: **Overview of TRANS-ZERO.** Once the tree is initiated, the search cycles the selection, expansion, simulation, and back-propagation for new nodes. b) G-MCTS selects the node with maximum UCB to expand a new child node. There are two types of genetic expansion: merge and mutate. c) A mass roll-out simulation of MTP trajectories assesses the semantic consistency. The assessed reward is backpropagated to guide the search. d) Finally, we harvest the search tree into data pairs for preference optimization.

maximum UCB node, are prepended to the instruction. The LLM then translates the **original input** x to the target language, with the given context.

- **Mutate.** We perform a mutation when the current maximum UCB node is the same as the maximum utility node. Mutation enables creative exploration based on existing translations, which is performed by translating a variant of the original input:

$$y_t = f(x' | \arg\max_{y<t}\{\text{UCB}(y)\}).$$

Specifically, the LLMs translate the **best** reconstruction $x' \in \{x_\omega\}$ recorded during simulations of the parent node, instead of the original input x:

The merge operation reduces the most promising MTP trajectories via an in-context translation of the original x. Meanwhile, the mutation extends the existing trajectory through additional reconstructed x' from simulation, which may come from languages other than source and target (§ 3.2 on simulations). Thus a search tree is constructed, with any path from the root to a node in the tree a valid MTP trajectory.

Simulation with Multilingual Semantic Consistency The simulation generates reward signals through semantic agreement across translations, which directly updates the UCB values in tree search to assess translation preferences without external supervision. The key insight is that a better translation candidate achieves higher semantic consistency for themselves and their descendants, thus guiding an optimized search.

As shown in Figure 2, assessing candidate y involves rolling out a temporary sub-tree on y through MTP with a width of b until it reaches a maximum depth of n. Each rollout step translates the parent node into a different language sampled from the supported languages $\{L_i\}$, with corre-

Utility (ν)	Tree Nodes	
0.5255	If you always harm others, the chickens gonna come home to roost.	
0.5595	① 如果你总是伤害别人，那么必然会有一天会有报应。	(If you always hurt others, there will be retribution one day.)
1.2290	② 你总是伤害别人，最后伤害的就是自己。	(You always hurt others, and in the end you end up hurting yourself.)
0.5801	③ 你总是伤害别人，最后伤害的还是你自己。	(You always hurt others, but in the end you end up hurting yourself.)
2.3230	④ 你总是伤害别人，最后总会有报应的。	(You always hurt others, and you will get your comeuppance in the end.)
0.6117	⑤ 你若总是伤害别人，鸡就要飞回你的巢了。	(If you keep hurting others, the chickens will fly back to your nest.)
0.5275	⑥ 如果你总是伤害别人，最后鸡蛋就要落回自己的头上。	(If you always hurt others, eventually the eggs will fall back on your head.)
0.5626	⑦ 你总是伤害别人，最后鸡蛋就要落回自己的头上。	(If you always hurt others, in the end the eggs will fall back on your head.)
0.5657	⑧ 如果你总是伤害别人，那么你也会遭到报应。	(If you always hurt others, then you will also suffer retribution.)
0.5601	⑨ 如果你总是伤害别人，那么最后总会有报应的。	(If you keep hurting others, you will get punished in the end.)
0.4676	⑩ 如果你总是伤害别人，那么麻雀总会飞回窝的。	(If you always hurt others, the sparrow will always fly back to the nest.)

Extracted Preference Pairs for Self-Play Preference Optimization (SPPO)

Chosen	Rejected	Win rates (softmax)
② 你总是伤害别人，最后伤害的就是自己。	① 如果你总是伤害别人，那么必然会有一天会有报应。	1.23 : 0.56 (0.6614)
④ 你总是伤害别人，最后总会有报应的。	② 你总是伤害别人，最后伤害的就是自己。	2.32 : 1.23 (0.7491)
④ 你总是伤害别人，最后总会有报应的。	③ 你总是伤害别人,最后伤害的还是你自己。	2.32 : 0.58 (0.8511)
⑦ 你总是伤害别人，最后鸡蛋就要落回自己的头上。	⑥ 如果你总是伤害别人，最后鸡蛋就要落回自己的头上。	0.56 : 0.52 (0.5088)
⑧ 如果你总是伤害别人，那么你也会遭到报应。	⑦ 你总是伤害别人，最后鸡蛋就要落回自己的头上。	0.57 : 0.56 (0.5008)

Table 1: **Example of Tree-to-Preference.** We perform level-order traversal of a search tree for English-to-Chinese translation. The first line presents the source input as the root. The Chinese translations are accompanied by corresponding English explanations enclosed in parentheses. Given that the utility near the root shall be larger, we apply a sorting algorithm to arrange the sequence in descending order, where each swap during the sort makes a preference pair.

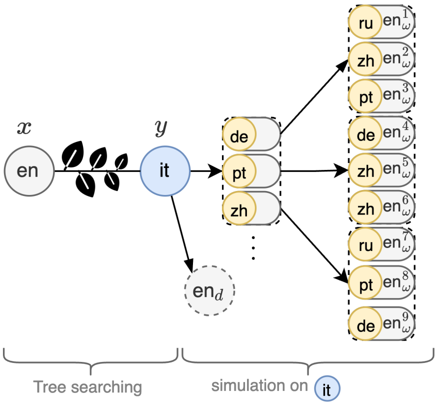

Figure 2: **An example of simulation on English-to-Italian translation candidate using $b = 3$ and $n = 2$.** Through roll-outs of MTP, the Italian candidate (ⓘⓣ) is assessed by semantics consistency of b^n English reconstructions $\{\text{en}_\omega^1, \cdots, \text{en}_\omega^9\}$ from simulated trajectories.

```
Algorithm 1 Tree-to-Preference Algorithm
Require: Translation candidates $\{y_*\}$ with utilities $\{\nu_*\}$, search tree $\mathcal{T}$ rooted from $x$
Ensure: Preference pairs with win rates for SPPO
  Step 1: Serialize and Sort Tree
  $\mathcal{S} \leftarrow$ LevelOrderTraversal($\mathcal{T}$)   ▷ Serialize tree and merge duplicates
  $\mathcal{S} \leftarrow$ SelectionSort($\mathcal{S}$, descending)   ▷ Sort nodes by utility
  Step 2: Generate Preference Pairs
  $\mathcal{P} \leftarrow \emptyset$
  for each swap $(y_i, y_j)$ in $\mathcal{S}$ do
      if $\nu_i > \nu_{\text{root}}$ and $\nu_i > \nu_j$ then
          win rate:$\mathbb{P} \leftarrow \frac{\exp(\nu_i)}{\exp(\nu_i)+\exp(\nu_j)}$
          $\mathcal{P} \leftarrow \mathcal{P} \cup \{(y_i \succ y_j, \mathbb{P}|x)\}$
      end if
  end for
  Return $\mathcal{P}$   ▷ Preference pairs with win rates for SPPO
```

sponding input reconstruction x_ω. Ultimately, this results in b^n MTP trajectories with reconstructions $\{x_\omega\}_{b^n}$. Given Eq. 4, the semantic consistency of these reconstructions $\{x_\omega\}_{b^n}$ with the original input x can be calculated as follows:

$$r(y) = \max(\underbrace{\overline{\mathrm{S}(x_\omega, x)}}_{\text{literal}}, \underbrace{\overline{\mathrm{S}(x_\omega, x_d)}}_{\text{free}}), x_\omega \in \{x_\omega\}_{b^n}$$

where x is the original input, and x_d is a direct reconstruction of the candidate y. Translation can be literal or free, with free translations assessed via straightforward back-translation x_d. Each assessment is *averaged* over all trajectories' $\{x_\omega\}_{b^n}$, with the superior one as the reward $r(y)$ derived from the simulation. The best simulation is recorded as x' for further expansions and simulations:

$$x' = \arg\max_{\{x_\omega\}_{b^n}} (S(x_\omega, x), S(x_\omega, x_d))$$

During the backpropagation, the reward $r(y)$ updates the utility ν and visit counts of all nodes on the trajectory from the current node back to the root according to Eq. 2.

3.3 Tree-to-Preference Algorithm

As Table 1 shows, once the G-MCTS is finished, we extract preference data from the translation candidates $\{y_*\}$ based on their utility $\{\nu_*\}$ by Algorithm 1. The tree is serialized where duplicate nodes are merged to their ancestor, with utilities and visit counts accumulated. Intuitively, nodes away from the root take more MTP steps to generate and thus face more risk of semantic loss as the translation step increases. Consequently, if a node has a higher utility than its ancestors or siblings, the corresponding translation is preferred for optimization. Furthermore, the root node tracks a comprehensive utility, which reflects the expected semantic consistency of the entire search. Thus, the utility of the preferred node shall also be higher than that of the root node. The preference pairs are utilized in SPPO, where their utilities are transformed into win rates through a softmax function. Note that translations that failed language detection may be generated during MTP, and their utility is halved as a penalty during the sorting and filtering.

4 Experiments

4.1 Settings

Data. We conduct experiments across six widely used languages: English (EN), German (DE), Portuguese (PT), Italian (IT), Chinese (ZH), and Russian (RU). We utilize the latest monolingual data from the WMT datasets. The SFT data for baselines is generated from the combination of the Flores-200 development set, with equal size for all translation directions. To evaluate multilingual translation performance, we employ the Flores-200 benchmark (Costa-jussà et al., 2022), assessing three key translation directions: (1) EN⇒X (English to other languages), (2) X⇒EN (other languages to English), and (3) X⇒X (inter-translations between non-English languages). These evaluations cover all translation directions of the six languages mentioned above.

Metrics. We evaluate translation quality with the reference-oriented metric BLEURT (Sellam et al., 2020) and reference-free metric COMET-KIWI (KIWI, Rei et al., 2022).

Baselines. We compare TRANS-ZERO with the following representative baselines:

- **General Instruct LLMs**: LLMs with off-the-shelf instruct-following ability for MT, e.g., Mixtral-8×7B, Llama3.1-8B-instruct and Qwen2.5-7B-instruct.
- **MT-oriented LLMs**: LLMs supervised by MT annotations, e.g., ALMA (Xu et al., 2024a) with parallel annotations, ALMA-R (Xu et al., 2024c) with preference annotations and Tower-Instruct (Colombo et al., 2024) with multi-task MT-related annotations, representing strong supervised baselines.
- **Base model and SFT model**: The base LLM for TRANS-ZERO, and their supervised counterparts.

Implementation. The training is conducted on 32 NVIDIA A100 GPUs (80GB). We utilize two base LLMs without instruction tuning: Llama-3.1-8b and Qwen-2.5-7b. Since some base LLMs (e.g., Llama-3.1-Base) lack the initial instruction-following for translation, we cold-start the LLM with a snippet of translation instructions (Appendix B). The TRANS-ZERO is parallelized across 32 threads, each assigned a batch of 10 sentences for random translation directions for G-MCTS. Upon completion of the search, we reduce all search threads for filtered preference data and apply Self-Play Preference Optimization (Chen et al., 2024, SPPO). Additional details are in Appendix B.

4.2 Main Results

As shown in Table 2, TRANS-ZERO based on Llama3.1 and Qwen2.5 achieves performance comparable to MT baselines trained on large-scale annotations, despite using only monolingual data for self-play. While TRANS-ZERO matches the English translation performance (EN⇒X) of ALMA-R, which also utilizes preference optimization, it significantly surpasses ALMA-R in non-English translation directions. Compared to Tower-instruct, an LLM trained on large-scale annotations, TRANS-ZERO exhibits slightly lower performance but remains highly competitive.

We also evaluated the translation performance of the base model and its instruction fine-tuned version for comparison. Through exploration learning, TRANS-ZERO significantly enhances the performance of both base models.

Additionally, we include SFT baselines using $5m$ parallel data by pairing the Flores200 development set, and cold-start instructions ($5k$) as parallel data. Although TRANS-ZERO does not match the performance by $5m$ supervision on EN⇒X and X⇒EN, it shows significant improvements in the non-English translation direction (X⇒X), achieving performance comparable to $5m$ supervision.

We further compared the performance improvement to varying scales of parallel annotations in

	EN⇒X		X⇒EN		X⇒X		Average	
	BLEURT	**KIWI**	**BLEURT**	**KIWI**	**BLEURT**	**KIWI**	**BLEURT**	**KIWI**
Mixtral-8x7B-Instruct	55.42	69.07	75.41	81.63	54.49	71.64	61.77	74.11
Llama3.1-Instruct	62.57	74.28	72.07	77.90	62.52	76.49	65.72	76.22
Qwen2.5-Instruct	72.16	81.99	77.70	84.60	68.59	79.87	72.82	82.15
ALMA	71.98	82.60	78.25	84.34	61.07	80.89	70.43	82.61
ALMA-R	69.38	83.10	77.52	84.87	51.03	82.47	65.98	83.48
Tower-Instruct	**76.74**	**85.26**	**78.73**	**85.02**	**72.98**	**83.08**	**76.15**	**84.45**
Llama3.1-Base	33.18	25.53	50.83	55.64	34.38	53.52	39.46	44.89
w/ SFT (40k)	74.46	82.30	77.30	84.03	71.23	80.02	74.33	82.12
w/ SFT (5m)	**75.80**	**84.61**	**78.47**	**84.61**	**73.30**	82.33	**75.86**	**83.85**
w/ **TRANS-ZERO**	73.71	83.20	77.60	84.34	73.28	**82.71**	74.86	83.42
Qwen2.5-Base	62.91	73.98	70.98	80.50	62.70	77.86	65.53	77.45
w/ SFT (5m)	**75.32**	**84.79**	**78.21**	**85.42**	**72.99**	**82.92**	**75.49**	**84.38**
w/ **TRANS-ZERO**	75.05	84.48	**78.21**	84.88	72.23	82.30	75.16	83.89

Table 2: **TRANS-ZERO achieves comparable and improved translation compared to SFT baselines with only monolingual self-play.** We highlight the best and the second-best performances in each section in **bold** and underlined, respectively.

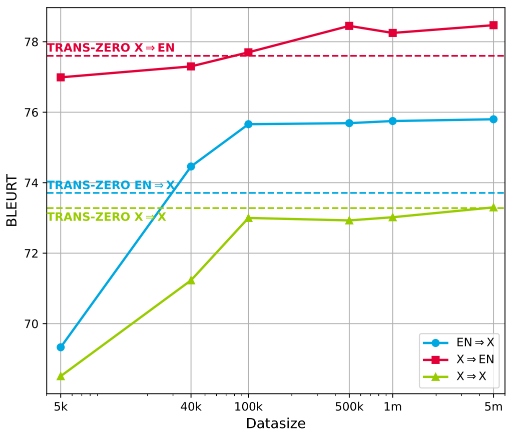

Figure 3: **BLEURT performance for SFT based on the Llama3.1-Base at different data sizes.** We include the performance of TRANS-ZERO in each language direction.

SFT. As shown in Figure 3, the translation quality saturates after more than 100k samples, especially for the EN⇒X and X⇒X directions. This suggests that simply increasing the amount of parallel annotations may not lead to proportional translation improvements.

Increasing the number of languages expands the potential exploration thus improving the TRANS-ZERO's performance upper bound. We explored with 4 and 6 languages for the G-MCTS. Figure 4 illustrates the performance changes in German-Chinese translation during Llama3.1-8b training. The number of languages used in tree search significantly impacts TRANS-ZERO's performance upper bound: increasing the number of languages can enhance the overall learning performance. With 6 languages, TRANS-ZERO essentially matches the performance of the open-source baseline system.

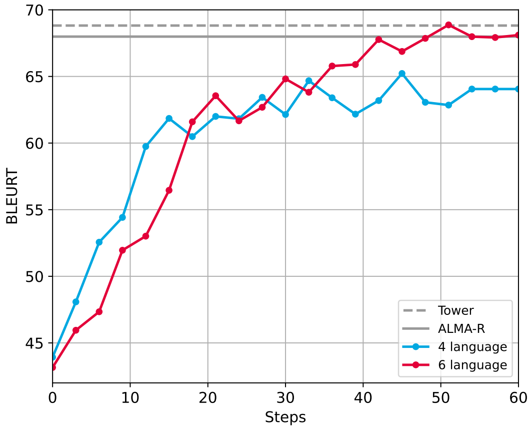

Figure 4: **The learning diagram of TRANS-ZERO on Llama3.1-Base for German-to-Chinese translation demonstrates the search process in 4-language and 6-language settings under G-MCTS.** By incorporating 6 languages, TRANS-ZERO attains BLEURT scores on par with the baseline systems.

4.3 Inference-time Scaling with G-MCTS

We further investigate the G-MCTS's potential to exploit high-quality translations as an inference-time scaling. Inference time scaling, such as Chain of Thought (CoT) (Wei et al., 2022b), has become popular for improving the performance of LLMs. Notably, traditional CoT requires additional learning given multiple natural language understanding supervisions. In contrast, G-MCTS enables straightforward inference-time scaling using only translation instructions (computation overheads detailed in the Appendix C). During the tree search, merging and mutation continuously explore and integrate relevant expressions from the multilingual

	EN⇒X		X⇒EN		X⇒X		Average	
	BLEURT	**KIWI**	**BLEURT**	**KIWI**	**BLEURT**	**KIWI**	**BLEURT**	**KIWI**
ALMA-R	69.38	83.10	77.52	84.87	51.03	82.47	65.98	83.48
+ G-MCTS	Failed	Failed	Failed	Failed	Failed	Failed	Failed	Failed
Tower-Instruct	76.74	85.26	78.73	85.02	72.98	83.08	76.15	84.45
+ G-MCTS	$76.44_{-0.30}$	$85.33_{+0.07}$	$78.28_{-0.45}$	$85.12_{+0.10}$	$\mathbf{74.42_{+1.44}}$	$83.57_{+0.49}$	$76.38_{+0.23}$	$84.67_{+0.22}$
Llama3.1-Base	33.18	25.53	50.83	55.64	34.38	53.52	39.46	44.89
+ G-MCTS	Failed	Failed	Failed	Failed	Failed	Failed	Failed	Failed
Llama3.1-Instruct	62.57	74.28	72.07	77.90	62.52	76.49	65.72	76.22
+ G-MCTS	$\mathbf{64.21_{+1.64}}$	$\mathbf{80.12_{+5.84}}$	$70.01_{-2.06}$	$\mathbf{79.86_{+1.96}}$	$\mathbf{68.12_{+5.60}}$	$77.12_{+0.63}$	$\mathbf{67.45_{+1.73}}$	$\mathbf{79.03_{+2.81}}$
Llama3.1-SFT (5k)	69.33	80.19	76.99	83.97	68.51	78.38	71.61	80.85
+ G-MCTS	$\mathbf{71.55_{+2.22}}$	$\mathbf{82.23_{+2.04}}$	$76.89_{-0.10}$	$84.00_{+0.03}$	$\mathbf{71.92_{+3.41}}$	$\mathbf{81.23_{+2.85}}$	$\mathbf{73.45_{+1.84}}$	$\mathbf{82.49_{+1.64}}$
Llama3.1-SFT (5m)	75.80	84.61	78.47	84.61	73.30	82.33	75.86	83.85
+ G-MCTS	$76.16_{+0.36}$	$84.95_{+0.34}$	$78.71_{+0.24}$	$84.68_{+0.07}$	$73.36_{+0.06}$	$82.48_{+0.15}$	$76.08_{+0.22}$	$84.04_{+0.19}$

Table 3: **G-MCTS enhances translation by scaling up inference, given the models' own instruction-following capability and multilingualism.** Performance improvements beyond one point are highlighted in **bold**. The base LLM and ALMA-R exhibit limitations due to their failure to follow instructions in various translation directions. The search particularly enhances X⇒X translations, where the availability of SFT annotations is significantly limited compared to English-related annotations.

semantic space, revising translations based on the LLM's conditional generation capabilities.

We employ G-MCTS with 6 languages to explore translations for the LLama-3.1 baselines, as well as Tower-Instruct and ALMA-R. The candidate of the highest utility makes the final translation. As Table 3 shows, G-MCTS requires a language model with basic instruction-following capabilities. Consequently, the Llama3.1-Base model fails the search due to its lack of instruction-following in various translation directions. Similarly, ALMA-R also fails due to its limited multilingualism, as indicated by its significantly lower X⇒X performance.

In contrast, Tower-Instruct and Llama3.1-Instruct significantly improve translation performance in the X⇒X direction, benefiting from multilingual priors. The base model trained with small-scale supervision also shows notable improvements. However, the improvement upon Llama3.1-SFT (5m) with large-scale supervision, is almost negligible. This suggests that when translations are fully activated, the performance gains from G-MCTS, rooted in LLM's inherent multilingualism, are not statistically significant.

5 Related Work

The utilization of LLM for machine translation has become popular, aligning with the prevailing trends in LLM applications. Xu et al. (2024a) first shifts the machine translation paradigm to fine-tuned LLM with moderate parallel supervision. Though LLM seems more data-efficient, it does not have a big appetite for large-scale supervision due to potential catastrophic forgetting (Xu et al., 2024a; Kondo et al., 2024). Directly scaling up multilingual MLE supervision hurts performance on resource-rich languages (Xu et al., 2024b). Therefore, XALMA (Xu et al., 2024b) hand-craft a mixture-of-expert to route different translation directions through separated modules, while ALMA-R (Xu et al., 2024c) turn to scale up more expensive preference tuning.

Preference tuning offers flexibility when fitting LLM with subjective human expectations for open-ended generations. However, the expense of preference annotation has led researchers to seek more cost-effective data sources, e.g., with additional assessments such as critic and revise modules (Huang et al., 2023; Tian et al., 2024; Zhang et al., 2024) Intuitively, the cross-lingual gaps in LLM offer a more scalable self-improvement preference as the proficient languages improve the lesser ones (Geng et al., 2024), e.g., a straightforward improvement roots in the direct mapping from the dominant linguistic ability as preference (She et al., 2024). Researchers further scale the preference by iterative and competitive gaming theory (Chen et al., 2024, SPPO), making it possible for models to self-improve. Recent work by Deepseek (Guo et al., 2025) has empirically validated its self-improving potential by employing large-scale reinforcement learning with its multilingual reasoning abilities.

6 Conclusion

In this work, we present TRANS-ZERO, a novel framework for multilingual machine translation that leverages multilingual LLMs with monolingual data only. Our experiments demonstrate that the proposed Genetic Monte-Carlo Tree Search (G-MCTS) effectively enhances translation quality by exploiting the LLM's inherent multilingual and instruction-following capabilities. Furthermore, we show that iterative training of G-MCTS, combined with preference optimization using monolingual data, TRANS-ZERO achieves scalable performance improvements, with the number of supported languages positively correlating with final translation quality. These findings establish a new direction for resource-efficient MT by shifting the paradigm from supervised parallel data to self-supervised monolingual learning.

Limitations

While TRANS-ZERO demonstrates promising results, it has several limitations that warrant discussion. First, the framework introduces higher computational overhead than supervised baselines, as the search process requires extensive exploration of the cross-lingual semantic space. Second, its effectiveness is inherently tied to the multilingual capabilities of the underlying LLM, rendering it less suitable for weaker models with limited cross-lingual alignment. Third, due to computational constraints, our experiments were limited in scale: we were unable to explore larger LLMs or extend the search process to a broader range of languages. Finally, the framework's performance upper bound may be influenced by the quality and diversity of monolingual data used for training, highlighting the need for future research into identifying the most cost-effective data types for self-supervised training.

Ethics Statement

The authors declare no competing interests. The datasets used in the training and evaluation come from publicly available sources and do not contain sensitive content such as personal information.

Acknowledgement

We would like to thank the anonymous reviewers for their insightful comments. Shujian Huang and Shanbo Cheng are the corresponding authors. This work is supported by the National Science Foundation of China (No. 62376116, 62176120), research project of Nanjing University-China Mobile Joint Institute, and the Fundamental Research Funds for the Central Universities (No. 2024300507).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *ArXiv preprint*, abs/2303.08774.

Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. 2012. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024. Self-play fine-tuning converts weak language models to strong language models. In *Proc. of ICML*. OpenReview.net.

Pierre Colombo, Duarte Alves, José Pombal, Nuno Guerreiro, Pedro Martins, Joao Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, et al. 2024. Tower: An open multilingual large language model for translation-related tasks. *ArXiv preprint*, abs/2402.17733.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *ArXiv preprint*, abs/2207.04672.

Xiang Geng, Ming Zhu, Jiahuan Li, Zhejian Lai, Wei Zou, Shuaijie She, Jiaxin Guo, Xiaofeng Zhao, Yinglu Li, Yuang Li, et al. 2024. Why not transform chat large language models to non-english? *ArXiv preprint*, abs/2405.13923.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *ArXiv preprint*, abs/2501.12948.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023. Large language models can self-improve. In *Proc. of EMNLP*, pages 1051–1068, Singapore. Association for Computational Linguistics.

Minato Kondo, Takehito Utsuro, and Masaaki Nagata. 2024. Enhancing translation accuracy of large language models through continual pre-training on parallel data. *ArXiv preprint*, abs/2407.03145.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, et al. 2024. Training language models to self-correct via reinforcement learning. *ArXiv preprint*, abs/2409.12917.

Jiahuan Li, Shanbo Cheng, Shujian Huang, and Jiajun Chen. 2024. MT-PATCHER: Selective and extendable knowledge distillation from large language models for machine translation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6445–6459, Mexico City, Mexico. 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.

Kumara Sastry, David Goldberg, and Graham Kendall. 2005. Genetic algorithms. *Search methodologies: Introductory tutorials in optimization and decision support techniques*, pages 97–125.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proc. of ACL*, pages 7881–7892, Online. Association for Computational Linguistics.

Shuaijie She, Wei Zou, Shujian Huang, Wenhao Zhu, Xiang Liu, Xiang Geng, and Jiajun Chen. 2024. MAPO: Advancing multilingual reasoning through multilingual-alignment-as-preference optimization. In *Proc. of ACL*, pages 10015–10027, Bangkok, Thailand. Association for Computational Linguistics.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.

Maciej Świechowski, Konrad Godlewski, Bartosz Sawicki, and Jacek Mańdziuk. 2023. Monte carlo tree search: A review of recent modifications and applications. *Artificial Intelligence Review*, 56(3):2497–2562.

Ye Tian, Baolin Peng, Linfeng Song, Lifeng Jin, Dian Yu, Lei Han, Haitao Mi, and Dong Yu. 2024. Toward self-improvement of llms via imagination, searching, and criticizing. In *Proc. of NeurIPS*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. of NeurIPS*, pages 5998–6008.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022a. Finetuned language models are zero-shot learners. In *Proc. of ICLR*. OpenReview.net.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Proc. of NeurIPS*.

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 *Proc. of ICLR*. OpenReview.net.

Haoran Xu, Kenton Murray, Philipp Koehn, Hieu Hoang, Akiko Eriguchi, and Huda Khayrallah. 2024b. X-alma: Plug & play modules and adaptive rejection for quality translation at scale. *ArXiv preprint*, abs/2410.03115.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. 2024c. Contrastive preference optimization: Pushing the boundaries of LLM performance in machine translation. In *Proc. of ICML*. OpenReview.net.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. 2024. Rest-mcts*: LLM self-training via process reward guided tree search. In *Proc. of NeurIPS*.

Dawei Zhu, Sony Trenous, Xiaoyu Shen, Dietrich Klakow, Bill Byrne, and Eva Hasler. 2024a. A preference-driven paradigm for enhanced translation with large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3385–3403, Mexico City, Mexico. Association for Computational Linguistics.

Wenhao Zhu, Hongyi Liu, Qingxiu Dong, Jingjing Xu, Shujian Huang, Lingpeng Kong, Jiajun Chen, and Lei Li. 2024b. Multilingual machine translation with large language models: Empirical results and analysis. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2765–2781, Mexico City, Mexico. Association for Computational Linguistics.

A Translation Prompts

TRANS-ZERO adopts all the mainstream translation instructions as prompts in Table 4. The G-MCTS adopts random instructions during sampling, and preference optimization adopts random instructions for the extracted preference pairs. LLMs for MT follow their default instructions for validation if available. TRANS-ZERO and other instruct-LLM baselines without default instruction adopt the ALMA instruction and <LABAL> as generation prompt for SFT and validation. All LLMs follow their default chat templates and generation prompts if available.

The translation context also samples instructions from Table 4 to organize translation pairs, which are then prepended to the translation instructions for in-context generation.

Model	Translation Instruction
ALMA & ALMA-R	Translate this from {src_lan} to {trg_lan}: \n{src_lan}: {src_sent} \n{trg_lan}:
Tower-Instruct	Translate the following text from {src_lan} into {trg_lan}.\n{src_lan}: {src_sent} \n{trg_lan}:
Others	Please translate the {src_lan} into {trg_lan}: {src_sent}
	{src_lan}: {src_sent} = {trg_lan}:
	{src_sent} in {src_lan} can be translated to {trg_lan} as:
	{src_lan}: {src_sent} \n {trg_lan}:
	Explain the following {src_lan} sentence in {trg_lan}: {src_sent}

Table 4: Commonly adopted translation instructions for LLM, {src_lan} and {src_lan} indicates the corresponding languages for source and target, and {src_sent} presents the input sentence of the source language.

B Implementation Details

For SFT baselines, we employ full parameter fine-tuning with a batch size of 1024.

For Llama3.1 cold-start, we generate approximately $5k$ translation instructions by **one random sentence tuple** from the Flores-200 development set, each representing one translation direction organized by a random instruction in Appendix A. The cold-start applies 1-epoch LoRA fine-tuning with rank $r = 64$ and scaling parameter $\text{Lora}_\alpha = 128$, optimized by AdamW at learning rate $lr = 1e - 4$.

For the G-MCTS, we configure a search width of $b = 5$ and a simulation depth of $n = 2$. The tree is fast-initiated with $b = 5$ child nodes on the root by sampling, with a maximum of 20 additional tree nodes per search. Note that the expansion in G-MCTS involves 6 times the sentence length during merging, while short sentences are not apt for diverse exploration, so we limit the G-MCTS to the sentence length within [30,256].

During a single self-play session, we distribute a monolingual batch across all devices, each launching a tree search for one sentence at a time, targeting a random translation direction. The search results are gathered from all devices, with preference pairs extracted and organized by sampled translation instructions (Table 4) for one SPPO epoch.

The SPPO follows all the parameter settings from DPO, including the coefficient $\eta = \frac{1}{\beta} = 10$, where β is the temperature parameter of DPO. The SPPO learning rate shall depend on the data size, where we adopt $lr = 1e - 6$ given approximately $25k$ preference pairs gathered, with the batch size of $10k$ preference pairs. One shall reduce the learning rate if more preference pairs are gathered during one self-play training epoch. Since preference extraction results may vary significantly on different sentences, we recommend scaling up the self-play session on a large sentence batch to ensure a stable training data flow.

The inference-time scaling with G-MCTS searches by the width of $b = 10$ given 6 languages (English, German, Portuguese, Italian, Chinese, and Russian) with simulation depth of $n = 2$.

C Computational Overhead of G-MCTS

The G-MCTS imposes substantial computational demands through its augmented inference architecture. Node-level operations exhibit significant latency differentials: merge and mutation operations require $6\times$ and $4\times$ prolonged in-context inference, respectively. Simulation for each node dynamically generates a subtree with b^n nodes. The self-play paradigm introduces multiplicative inference loads, where each monolingual training sample necessitates $20 \times 5^2 = 500$ times more inference, while the inference-time scaling in our experiments introduces $20 \times 10^2 = 2000$ times more inference.