Learning to Translate Ambiguous Terminology by Preference Optimization on Post-Edits

Nathaniel Berger[§*], **Johannes Eschbach-Dymanus**[‡], **Miriam Exel**[‡]
Matthias Huck[‡], **Stefan Riezler**[†§]
[§]Computational Linguistics & [†]IWR, Heidelberg University, Germany
[‡]SAP SE, Dietmar-Hopp-Allee 16, 69190 Walldorf, Germany
{berger, riezler}@cl.uni-heidelberg.de
{johannes.eschbach-dymanus, miriam.exel, matthias.huck}@sap.com

Abstract

In real world translation scenarios, terminology is rarely one-to-one. Instead, multiple valid translations may appear in a terminology dictionary, but correctness of a translation depends on corporate style guides and context. This can be challenging for neural machine translation (NMT) systems. Luckily, in a corporate context, many examples of human post-edits of valid but incorrect terminology exist. The goal of this work is to learn how to disambiguate our terminology based on these corrections. Our approach is based on preference optimization, using the term post-edit as the knowledge to be preferred. While previous work had to rely on unambiguous translation dictionaries to set hard constraints during decoding, or to add soft constraints in the input, our framework requires neither one-to-one dictionaries nor human intervention at decoding time. We report results on English-German post-edited data and find that the optimal combination of supervised fine-tuning and preference optimization, with both term-specific and full sequence objectives, yields statistically significant improvements in term accuracy over a strong translation oriented LLM without significant losses in COMET score. Additionally, we release test sets from our post-edited data and terminology dictionary.

1 Introduction

In business scenarios, accurate terminology translation is critical to ensure that the translated text is understood as intended. Ambiguous terminology can make this more difficult. Take, for example, the term 'transfer' (Figure 1). In German, our terminology dictionary specifies 27 possible translations of the term 'transfer', depending on the context and part of speech. A 'transfer' could be a delivery, in which case it is a 'Übergabe' or 'Warenüberführung', whereas on the other hand, it could be a transfer of data, which makes it a 'Übertragung'. If you are calling a company on the phone and they have to transfer you to another department, it is a 'Weiterleitung'. Working with a logistics company, any of these could be possible translations of 'transfer'. Neural machine translation (NMT) is quite capable of using sentence level context to disambiguate terminology, but it is not perfect and frequently human translators are required to intervene and correct machine translations to create post-edited translations. However, even though the professional translators who perform post-editing are knowledgeable about how to translate terms depending on their usage, which term translation they use varies from editor to editor. Moreover, if NMT is provided to end users, it faces the challenge that users may not know the correct term translations or even speak one of the languages. This makes approaches that require annotating source terms with the desired target translation less tenable. Automated solutions are possible if an unambiguous dictionary is available, or else a system could detect when the source text contains a term and annotate it with all possible term translations, but it still is the responsibility of the NMT system to determine

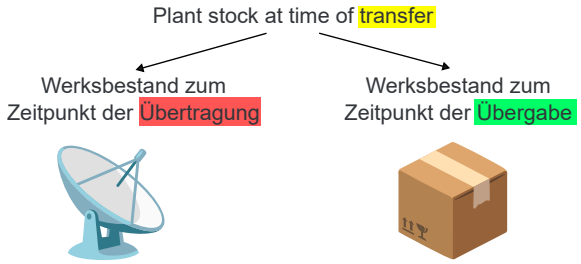

Figure 1: The term 'transfer' is a highly ambiguous term in our dictionary with 27 possible translations. Transfer could be translated to "Übertragung" which would refer to a transfer of data or a broadcast. On the other hand, "Übergabe" could mean a transfer of physical goods. In this case, the source sentence is likely referring to physical goods with "Plant stock".

[*]Work was done prior to joining Amazon.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 607–617
November 4-9, 2025

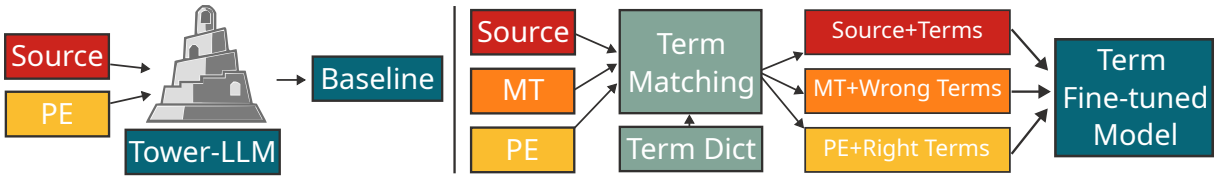

Figure 2: In order to train terminology corrected models, we begin by performing continued pre-training on Tower-7B base on in-house post-edits to serve as our baseline. We then match our terminology in post-edits and MT to find MT with incorrect but valid terms and post-edits with corrected terms. This is used to train our baseline for term fine-tuned models.

the correct translation.

This work is a step towards an NMT system that is more capable of producing correct term translations without the need for a one-to-one terminology dictionary, or for human intervention at decoding time. The central idea is to to learn terminology translation from statistical information about context and editor preferences that is encoded in post-edits. We show that a combination of supervised fine-tuning and preference optimization (PO) for knowledge editing on (post-edit, machine translation) pairs (Rozner et al., 2024; Berger et al., 2024) with term-specific and sequence level objectives, the capabilities of NMT to disambiguate depending on context knowledge can be significantly improved by exploiting fine-grained discriminative information about preferred and dispreferred term translations. We create a variant of IPO (Gheshlaghi Azar et al., 2024) and of supervised fine-tuning (SFT) that masks non-term tokens. We then create combinations of masked and unmasked losses to investigate what benefits ambiguous term translation. Our experimental results show that PO learns a margin between preferred and dispreferred sequences, with the effect that semantically similar term translations are separated and the correct terminology variant is selected. Evaluation scores on English-German post-edited data show that the optimal combination of SFT and PO, where we combine a masked objective to target terminology with training on full sequences, yields statistically significant improvements in term accuracy over a strong baseline LLM for translation (Tower Base LLM 7B) pre-trained on in-domain data, without significant losses in COMET score. Translation with foundational LLMs (GPT 4.1) yields competitive COMET scores, but cannot exploit ambiguous term dictionaries profitably.

A further contribution of our work is a release of terminology annotated test sets, containing source text, machine translations, and post-edits. Half of each test set contains ambiguous terms where the MT uses an incorrect term variant and the PE corrects term in the MT.

2 Related Work

Previous works on terminology in translation have focused on adding either hard constraints on decoding, such as in constrained beam-search (Hokamp and Liu (2017), Post and Vilar (2018)), or adding a soft constraint by providing the source and target term pair as an additional model input (Dinu et al. (2019), Exel et al. (2020)). Adding the terminology translation as an extra input has been expanded upon with the rise of LLMs and in-context learning to prompt-based methods (Moslem et al., 2023). The latter approach was the most popular in the latest iteration of the WMT shared task on terminology translation (Semenov et al., 2023), with all six of the evaluated systems using terms as an additional input and a few adding either a post-processing step or modifying decoding strategies to further enforce term usage. With both hard-constraints during decoding or soft-constraints in the input, one most know beforehand what the correct term translation is. For term dictionaries that contain a one-to-one term mapping, this is no problem. However, when the term dictionary contains a one-to-many mapping, the decision of what term translation is correct is offloaded from the system to a human annotator or translator.

Work has been done to add more advanced fulfillment criteria to the constrained beam search that allow multiple variants of a term to fulfill the constraint. For example Hauhio and Friberg (2024), use finite state automata to check for constraint fulfillment and to recognize all acceptable term inflections. However, in order to only accept a specific term translation, one would again have to manually specify what the desired translation is.

Our approach is based on preference optimization for text generation models (Christiano et al.,

2017; Rafailov et al., 2023; Gheshlaghi Azar et al., 2024; Xu et al., 2024) applied to performing knowledge editing (Rozner et al., 2024) on our baseline model. We begin with a baseline model that is already quite capable of producing valid terms that exist in our term dictionary. From the possible valid terms, we would like to produce exactly *the* correct term and not *an* extant term.

3 Methods

Our task varies from the WMT terminology translation task in the following ways: first, the terminology dictionaries used to evaluate the shared task were created by aligning source and reference texts and having human annotators correct false alignments or incorrect terminology. Second, the segment level term dictionaries were provided to the NMT system at evaluation time. Segment level term dictionaries created by alignment ensure that, for a term in a given segment, there is only one correct translation.

We focus on applying an existing terminology dictionary with ambiguous terms, containing multiple translations for a given source term, by using terminology focused loss functions. Additionally, we do not provide the term dictionary as an extra input to the model next to the source segment, as Moslem et al. (2023) do. In order to tackle our problem of ambiguous terminology, we need to find examples of terms being incorrectly used and then corrected, and training objectives that make use of the pair of (correct, incorrect) terminology. An overview of our training process is illustrated in Figure 2.

3.1 Terminology Matching

To find incorrect terms in our machine translations and their corrections in our post-edits, we use fuzzy matching. We use the RapidFuzz library[1] to calculate fuzzy matching ratios and align the terms in the dictionary and text. This is done with the function `partial_ratio_alignment`, which returns a span in the text and a score. The fuzzy match score is calculated as 1 minus the normalized Levenshtein distance, with substitution cost of 2. The score threshold is set to 0.95 to allow minor variations.

First, we check if the source text contains a valid term. If it does, we then look for all possible term translations in both the MT and PE. If both the MT and PE contain terms, we then make sure that they don't contain the same term. Because some term translations overlap (both 'transfer' and 'transferieren' would match if the latter were contained in the text), we may have a set of matches for MT or PE. We accept the term matches if the intersection of the MT and PE term match sets is empty.

As some terms are subsequences of other terms, i.e. "Überführung" is wholly contained within "Warenüberführung", we remove overlapping terms by checking if one matched term is contained within another matched term. If that is the case, then we take the larger of the two matches.

3.2 Training Objectives

Our training objective modifies the dCPO loss of Berger et al. (2024) for preference optimization on terminology-containing post-edit pairs. The dCPO loss is a direct preference optimization loss based on the alternative formulation of preference optimization by Gheshlaghi Azar et al. (2024), which introduced the IPO loss, combined with the insight of Xu et al. (2024) that adding an SFT term improved the ability of the language model to learn the reference translations. It is defined as

$$\mathcal{L}_{dCPO}(y_w, y_l, x) = -\log(\pi^*(y_w|x)) + \left(\left(\log\frac{\pi^*(y_w|x)}{\pi_{ref}(y_w|x)} - \log\frac{\pi^*(y_l|x)}{\pi_{ref}(y_l|x)}\right) - \frac{1}{2\beta}\right)^2 \quad (1)$$

where π^* is the model currently being trained, π_{ref} is the initial model, y_w is the preferred sequence, and y_l is the dis-preferred sequence. $\pi(y|x)$ is calculated as

$$\sqrt[|y|]{\Pi_{i=0}^{|y|}\pi(y_i|y_{<i}, x)}$$

which is the geometric mean of target token-level probabilities, the log of which is proportional to the arithmetic mean of log-probabilties as described in Berger et al. (2024). The π_{ref} in the denominator serves as a KL-divergence regularizer on the loss so that baseline performance is not lost. However, because we are focused on fixing ambiguous terms in our baseline model, we do not want to regularize towards this behavior and therefore remove it, similar to Xu et al. (2024).

During training, we noticed that the SFT term contained in the dCPO loss can easily be overwhelmed by the IPO term when the distance set in the IPO term is much larger than the initial distance between sequences in log-probability space.

[1] https://github.com/rapidfuzz/RapidFuzz

Because IPO is based on a squared-error loss, gradients grow multiplicatively with the error. This is problematic if the baseline model has already fit the post-edits well, meaning the SFT term's gradient has a fairly small norm in comparison to the IPO term. To ameliorate this problem, we introduce two modifications. First, we replace the squared-error component of the loss with smooth-l1.

$$sl_1(x, y) = \begin{cases} 0.5(x-y)^2 & \text{if } |x-y| < 1 \\ |x-y| - 0.5 & \text{otherwise} \end{cases} \quad (2)$$

This removes the multiplicative growth of the gradient with regard to the input, to reduce the exploding gradient problem. Thus our preference optimization loss becomes

$$\mathcal{L}_{PO}(x, y_w, y_l) = sl_1\left(\log(\pi^*(y_w|x)) - \log(\pi^*(y_l|x)), \tfrac{1}{2\beta}\right) \quad (3)$$

Our second addition is to simply add a weight α on the SFT loss term to attempt balancing it with the PO loss term.

$$\mathcal{L}_{SFT}(y, x) = -\alpha \log(\pi^*(y|x)) \quad (4)$$

In order to introduce preferences more fine-grained than the sequence level to the model, we introduce non-term token masking into the loss. To do this, we construct a set of token indices, and compute

$$\tilde{\pi}_\delta(y|x) = \sqrt[|\delta|]{\Pi_{i \in \delta} \pi(y_i|y_{<i}, x)}$$

with all $i \in \delta$ being indices into the sequence y such that y_i is part of a term.

We then create a masked variant of both the preference optimization and SFT losses, with masked preference optimization (mPO) being

$$\mathcal{L}_{mPO}(x, y_w, y_l, \delta_w, \delta_l) = sl_1\left(\log(\tilde{\pi}^*_{\delta_w}(y_w|x)) - \log(\tilde{\pi}^*_{\delta_l}(y_l|x)), \tfrac{1}{2\beta}\right) \quad (5)$$

and masked supervised fine-tuning (mSFT) as

$$\mathcal{L}_{mSFT}(x, y_w, \delta_w) = -\alpha \log(\tilde{\pi}^*(y_w|x, \delta_w)) \quad (6)$$

These variants can be combined with each other such that we can perform preference optimization across entire sequences with extra emphasis on the terminology tokens. Or we could perform supervised fine-tuning on only terms.

We assign an indicator to each loss component to switch it on and off for different training runs. This results in our final loss function,

$$\mathcal{L}_{term} = \mathbb{1}_{PO}\mathcal{L}_{PO} + \mathbb{1}_{mPO}\mathcal{L}_{mPO} + \alpha(\mathbb{1}_{SFT}\mathcal{L}_{SFT} + \mathbb{1}_{mSFT}\mathcal{L}_{mSFT}) \quad (7)$$

a combination of loss functions (3), (4), (5), and (6), which we evaluate with multiple settings to find the best for learning ambiguous terms.

4 Experiments

We perform two sets of experiments, a set of prompting experiments to serve as a baseline and fine-tuning experiments with variations of the $\mathcal{L}_{term}$ loss. For our prompting experiments, we use GPT 4.1[2] from OpenAI. The prompting experiments were performed to see if commercial LLMs can perform the term disambiguation task without any training. We prompt with and without the term dictionary for the given source segment. Prompt templates for this experiment are available in Appendix D. Results for both the prompting and fine-tuning experiments are shown in section 5, in Tables 1 and 2, respectively.

4.1 Data

We begin with a large corpus of post-edits on English to German machine translations produced by multiple NMT systems over multiple years. From our corpus of post-edits, we select examples of machine translations and post-edits containing ambiguous terminology with the method outlined in section 3.1 to create a training set of 123,518 examples. This data contains only examples where the machine translation contains a term translation in our dictionary, but the post-editor corrected it to a different translation from the dictionary. We additionally select 6000 examples for validation, and 2000 for testing. Our validation and test sets have evenly balanced terminology and non-terminology subsets. Our training data contains 3579 unique source terms. On average, each ambiguous term in our term dictionary has 3.32 possible translations with a standard deviation of 1.85. The most extreme case was the term 'transfer' with 27 possible translations in our dictionary. In our test set, we find 335 unique source terms with 4.89 possible translations on average. Figure 3 in Appendix B shows a histogram of the target term counts. The

[2] gpt-4.1-2025-04-14

Model	ChrF	COMET	Term Accuracy
GPT 4.1 w Terms	69.5	82.24	37.1
GPT 4.1 w/o Terms	69.7	82.58	43.5

Table 1: Results from prompting experiments to serve as a baseline. GPT 4.1 refers to gpt-4.1-2025-04-14.

training data terminology has high coverage of the test set, with 97.8% of terms in the test data also appearing in the training data. On our test set, we calculate the term accuracy that could be achieved by a random baseline as 24.9%. The random baseline is simply a random choice over all possible term translations given the source term.

4.2 Model Training

We begin with Tower Base LLM 7B (Alves et al., 2024) and evaluate it without any additional training on our test set to establish its performance. Initially, it is translating 36.1% of the terms correctly, achieving a COMET score of 78.72 and a ChrF score of 63.9. We then perform continued pre-training on the 123,518 English-German training examples to adapt it to our domain and to begin learning our terminology—the resulting model serves as our baseline LLM.

Hyperparameters for the continued pre-training step and further fine-tuning can be found in Appendix A. For all of our trainings we perform full fine-tuning using Accelerate[3] with distributed data-parallelism. For our baseline model we use COMET score on the validation set as an early stopping mechanism.

From this initial training, the model already learns our term dictionary quite well. 94.4% of the time, it is able to correctly predict a valid term in our dictionary when faced with an ambiguous source term. However, that does not necessarily mean it is getting exactly the correct term. When we check for the exact term used in the test data, we find that only 53.7% of the time is it picking the correct term translation. Additionally, it produces the exact same erroneous terminology choice that appears in the original machine translations 34.4% of the time, suggesting that the knowledge-editing approach with preference optimization is appropriate.

Once we have our baseline LLM, we perform fine-tuning for our term disambiguation task with various configurations of $\mathcal{L}_{term}$. Instead of using COMET as an early stopping criterion, we now use term accuracy only.

We consider 6 different settings of $\mathcal{L}_{term}$. Setting 1 has only $mSFT$ and 2 has $SFT + mSFT$ enabled. Settings 3 through 6 then contain different combinations of preference optimization and supervised fine-tuning.

Setting 3 examines $SFT + PO$, which is similar to the dCPO set-up of Berger et al. (2024). Setting 4 considers only the masked variants, so $mSFT + mPO$, to examine if only optimizing the term tokens suffices. Setting 5 uses $SFT + mPO$ to ensure that the entire post-edit remains likely while attempting preference optimization on terms only. Setting 6 combines all objectives $SFT + mSFT + PO + mPO$ to see if the objectives are complementary.

4.3 Metrics

Following the previous shared task for terminology translation at WMT (Semenov et al., 2023), we evaluate translations produced by our models with COMET (Rei et al., 2022), ChrF (Popović, 2015), and term accuracy.

The COMET model we use is based on the direct-assessment variant with references[4] but fine-tuned on our own direct assessment data. Zouhar et al. (2024) show that fine-tuned neural metrics, such as COMET, show worse correlation with human judgements on out of domain data. We find that our fine-tuned COMET model has higher correlations with human judgements on our internal domains than the version released by Rei et al. (2022).

In accordance with our data pre-processing, we also perform fuzzy matching with terminology to evaluate our term accuracy. We search for terminology in translation outputs with a threshold of 0.95 to allow for minor variations, such as different inflections for verb forms or plurals for nouns. If a match is found within this threshold, we count it towards our term accuracy.

[3] https://github.com/huggingface/accelerate

[4] https://huggingface.co/Unbabel/wmt22-comet-da

Init	Model	SFT	mSFT	PO	mPO	α	ChrF	COMET	Term Accuracy
Tower	Baseline	✓	✗	✗	✗	1	73.5	83.14	53.7
Baseline	1	✗	✓	✗	✗	1	72.3	82.74	56.1
Baseline	2	✓	✓	✗	✗	1	73.2	83.01†	55.8
Baseline	3	✓	✗	✓	✗	1	72.2	82.65	54.6
Baseline	4	✗	✓	✗	✓	10	72.0	82.62	55.9
Baseline	5	✓	✗	✗	✓	10	73.5	83.15†	55.6*
Baseline	6	✓	✓	✓	✓	10	73.0	83.03†	56.3*

Table 2: Results for rebooted experiments with Baseline trained only on the term data. COMET results marked with † have *not* significantly changed from the baseline. Term Accuracy results marked with a $*$ *have* significantly improved over the baseline.

5 Results

In Table 1, we report ChrF, COMET, and term accuracy for the GPT 4.1 prompting experiments. GPT 4.1, without the terminology, translates quite well, achieving a COMET score of 82.58. Without the term dictionary, it achieves a term accuracy of 43.5%. If we consider any possible term translation and not only the correct term translation, then GPT 4.1 without the dictionary would achieve a term accuracy of 86.9%. When given the term dictionary, its ability to use the correct term deteriorates, dropping to 37.1%. However, if we consider again term accuracy over any possible term translation, then it would achieve a term accuracy of 98.2%. This suggests that while prompting might be a straightforward way to use a dictionary, an LLM is not quite capable of disambiguating our terminology. This further motivates a fine-tuning approach.

In Table 2 we report the term accuracy, ChrF and COMET scores for our all of our trained models. For our fine-tuning runs using $\mathcal{L}_{term}$, we report the setting of our indicators $\mathbb{1}$ with ✓and ✗.

Our baseline system achieves a term accuracy of 53.7%, better than the 24.9% we would expect from a random choice of term translations and the GPT 4.1 result of 43.5%. This baseline is then used to initialize all following experiments in Table 2.

When fine-tuning our baseline model, we see that all settings of $\mathcal{L}_{term}$ positively affect term accuracy, with the best setting being all indicators set to 1 (model 6). This achieves a term accuracy of 56.3%. The combination of $\mathcal{L}_{SFT} + \mathcal{L}_{mSFT}$ also brings improvements and achieves a term accuracy of 55.6%. Settings 5 and 6 achieve significant improvements over the baseline, with $p < 0.05$ according to a pairwise approximate randomization significance testing (Riezler and Maxwell, 2005). Although there are other results that achieve higher term accuracy, their lack of significance suggests that these results have higher variance.

Regarding machine translation quality, we see that settings 5 and 6 yield COMET scores that are not significantly different from the baseline. Some training settings see degradation with regards to ChrF and COMET score while making gains in term accuracy, specifically settings 1, 3, and 4. This is seen starkly in settings that do not see the entire sequences but rather just the terms. Settings 1 and 4, consisting of $mSFT$ and $mSFT + mPO$ respectively, lose 1.2 and 1.5 ChrF points and 0.40 and 0.52 COMET points. This loss in COMET is a significant difference from the baseline model. This suggests that focusing only on terminology can cause the model to begin forgetting some more general translation abilities. However, settings 2, 5, and 6 do not see significant losses in terms of COMET score. Notably, all of these settings contain $\mathcal{L}_{SFT}$, indicating the continuing to train with SFT on the full sequence is necessary to mitigate losses while still gaining term accuracy.

6 Conclusion

We introduced a preference optimization-based approach to improve terminology disambiguation in neural machine translation. By leveraging post-edited corrections of ambiguous terms, we trained models to better select the correct translation from multiple valid options.

Our best-performing model (setting 6: $\mathcal{L}_{SFT} + \mathcal{L}mSFT + \mathcal{L}_{PO} + \mathcal{L}mPO$) improved term accuracy from 53.7% to 56.3%. These gains come without significant losses on COMET, while models trained on term-focused losses alone saw larger degradation in terms of COMET and ChrF. This suggests that term-focused optimization can shift

model behavior in ways that improve terminology but only insignificantly reduce overall fluency (Gisserot-Boukhlef et al., 2024).

One hypothesis for the trade-offs is that semantically similar term translations, such as Übergabe and Übernahme for transfer, are likely close in embedding space, making it difficult for the model to separate them. If PO forces the model to distinguish between highly related terms, it may unintentionally alter how word choices are processed—leading to losses in general translation quality. Because of this, it is necessary to continue supervised fine-tuning on full sequences.

Acknowledgements

The last author acknowledges support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant INST 35/1597-1 FUGG.

Limitations

Limitations of our work include a broader evaluation on more language pairs. This shortcoming is due to the size of our post-editing data, which is largest for the English-German language pair, while post-edits for other language pairs are too small for training purposes. Furthermore, alternative preference optimization techniques, e.g., reinforce-style (Ahmadian et al., 2024) with verifiable rewards on terminology fuzzy matches, similar to Lambert et al. (2024) and Rei et al. (2025), could also be applied to this task and serve as a comparison point for our methods.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. 2024. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, Bangkok, Thailand.

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

Nathaniel Berger, Miriam Exel, Matthias Huck, and Stefan Riezler. 2024. Post-edits are preferences too. In *Proceedings of the Ninth Conference on Machine Translation*, pages 1289–1300, Miami, Florida, USA. Association for Computational Linguistics.

Paul F. Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems (NIPS)*, Long Beach, CA, USA.

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. Training neural machine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy. Association for Computational Linguistics.

Miriam Exel, Bianka Buschbeck, Lauritz Brandt, and Simona Doneva. 2020. Terminology-constrained neural machine translation at SAP. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 271–280, Lisboa, Portugal. European Association for Machine Translation.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 4447–4455. PMLR.

Hippolyte Gisserot-Boukhlef, Ricardo Rei, Emmanuel Malherbe, Céline Hudelot, Pierre Colombo, and Nuno M. Guerreiro. 2024. Is preference alignment always the best option to enhance LLM-based translation? an empirical analysis. In *Proceedings of the Ninth Conference on Machine Translation*, pages 1373–1392, Miami, Florida, USA. Association for Computational Linguistics.

Iikka Hauhio and Théo Friberg. 2024. Mitra: Improving terminologically constrained translation quality with backtranslations and flag diacritics. In *Proceedings of the 25th Annual Conference of the European Association for Machine Translation (Volume 1)*, pages 100–115, Sheffield, UK. European Association for Machine Translation (EAMT).

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Nathan Lambert, Jacob Daniel Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang,

Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hanna Hajishirzi. 2024. Tülu 3: Pushing frontiers in open language model post-training. *ArXiv*, abs/2411.15124.

Yasmin Moslem, Rejwanul Haque, John D. Kelleher, and Andy Way. 2023. Adaptive machine translation with large language models. In *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, pages 227–237, Tampere, Finland. European Association for Machine Translation.

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.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

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, Nuno M. Guerreiro, José Pombal, João Alves, Pedro Teixeirinha, Amin Farajian, and André F. T. Martins. 2025. Tower+: Bridging generality and translation specialization in multilingual llms. *Preprint*, arXiv:2506.17080.

Stefan Riezler and John T. Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 57–64, Ann Arbor, Michigan. Association for Computational Linguistics.

Amit Rozner, Barak Battash, Lior Wolf, and Ofir Lindenbaum. 2024. Knowledge editing in language models via adapted direct preference optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4761–4774, Miami, Florida, USA. Association for Computational Linguistics.

Kirill Semenov, Vilém Zouhar, Tom Kocmi, Dongdong Zhang, Wangchunshu Zhou, and Yuchen Eleanor Jiang. 2023. Findings of the WMT 2023 shared task on machine translation with terminologies. In *Proceedings of the Eighth Conference on Machine Translation*, pages 663–671, Singapore. Association for Computational Linguistics.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. 2024. Contrastive preference optimization: pushing the boundaries of llm performance in machine translation. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Vilém Zouhar, Shuoyang Ding, Anna Currey, Tatyana Badeka, Jenyuan Wang, and Brian Thompson. 2024. Fine-tuned machine translation metrics struggle in unseen domains. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 488–500, Bangkok, Thailand. Association for Computational Linguistics.

A Hyperparameters

Hyperparameter	Value
Max Epochs	20
Learning Rate	1e-5
Optimizer	AdamW
Learning Rate Scheduler	Cosine
Warm-up Ratio	0.05
Effective Batch Size	256
Max Gradient Norm	10.0
Mixed Precision	`bfloat16`
Early Stopping Criterion	Internal COMET
Early Stopping Patience	3
Early Stopping Epsilon	0.00001
Evaluation Frequency	1000 steps
Max New Tokens	64
Average Log-Probabilities	`True`
Normalize Loss	`True`

Table 3: Hyperparameters for our baseline model with continued pre-training on all post-edit data.

Hyperparameter	Value
Max Epochs	20
Learning Rate	2e-6
Optimizer	AdamW
Learning Rate Scheduler	Cosine
Warm-up Ratio	0.05
Effective Batch Size	256
Max Gradient Norm	1.0
Mixed Precision	`bfloat16`
Early Stopping Criterion	Term Accuracy
Early Stopping Patience	3
Early Stopping Epsilon	0.00001
Evaluation Frequency	250 steps
Max New Tokens	64
β	0.25
Average Log-Probabilities	`True`
Normalize Loss	`True`

Table 4: Hyperparameters for fine-tuning on terminology containing data subset.

B Term Ambiguity

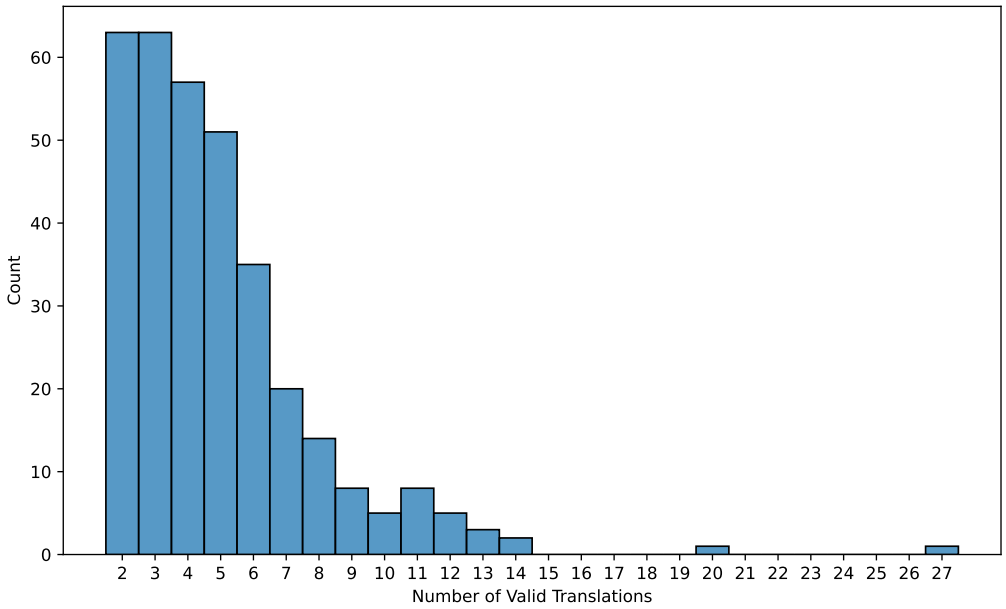

Figure 3: Here we show a histogram of the number of unique source terms in the test set for a given amount of valid translation terms. For the 335 unique source terms in the test set, on average they have 4.89 possible translations with a standard deviation of 2.97.

C Sample dictionary entry

`transfer -> {Versetzung, weitergeben, Warenbewegung, umlagern, Verlegung, übernehmen, Raumüberwindung, übertragen, Weiterleitung, Übertragung, Umbuchung, verlegen, Warenüberführung, Umladung, abführen, umbuchen, Versendung, Umlagerung, Übernahme, überleiten, Transfer, weiterleiten, Überführung, transferieren, Übergabe, Überleitung, Verfügung}`

Figure 4: Mapping from a source term, 'transfer', to multiple valid target terms. Our term dictionary is a one-to-many mapping. Some target terms are also semantically overlapping, causing difficulties for the language model to produce the correct term translation.

D Prompt Templates

```
[
  {
    "role": "system",
    "content": "You will be provided with a user input in English. Translate the text into German.

    The translation must satisfy the following terminology constraints:
    reference quantity - Bezugsmenge, Referenzmenge, Preiseinheit
    If more than one translation is possible for a given term, please select the best term.

    Only output the translated text, without any additional text."
  },
  {
    "role": "user",
    "content": "You can display the reference quantity and the simulation quantity side by side."
  }
]
```

```
[
  {
    "role": "system",
    "content": "You will be provided with a user input in English. Translate the text into German.

    Only output the translated text, without any additional text."
  },
  {
    "role": "user",
    "content": "You can display the reference quantity and the simulation quantity side by side."
  }
]
```

Figure 5: Sample prompts for GPT 4.1. The upper prompt shows the added terminology constraints while the lower prompt is without constraints.