Lingua Custodia's participation at the WMT 2023 Terminology shared task

Jingshu Liu **Mariam Nakhlé** **Gaëtan Caillaut** **Raheel Qader**
Lingua Custodia, France
{jingshu.liu,mariam.nakhle,gaetan.caillaut,raheel.qader}@linguacustodia.com

Abstract

This paper presents Lingua Custodia's submission to the WMT23 shared task on Terminology shared task. Ensuring precise translation of technical terms plays a pivotal role in gauging the final quality of machine translation results. Our goal is to follow the terminology constraint while applying the machine translation system. Inspired by the recent work of terminology control, we propose to annotate the machine learning training data by leveraging a synthetic dictionary extracted in a fully non supervised way from the give parallel corpora. The model learned with this training data can then be then used to translate text with a given terminology in a flexible manner. In addition, we introduce a careful annotated data re-sampling step in order to guide the model to see different terminology types enough times. In this task we consider all the three language directions: Chinese to English, English to Czech and German to English. Our automatic evaluation metrics with the submitted systems show the effectiveness of the proposed method.

1 Introduction

It is well proven that modern Neural Machine Translation (NMT) systems (Sutskever et al., 2014; Bahdanau et al., 2014; Luong et al., 2015; Vaswani et al., 2017) achieve generally satisfying translation results. Nonetheless, the performance of translation with terminology control remains to be improved. This paper describes our submission to the WMT23 Terminology translation task in Chinese to English, English to Czech and German to English direction. The task aims to develop and evaluate machine translation systems which can translate domain specific terms in an accurate and consistent way with some extra terminology information. Note that the terminology is provided only in the inference phase, for the training there's no existing resources about the terminology.

Previous works on machine translation with terminology control can be grouped into two categories according to whether the method needs training the model with terminology information. One group incorporates the constraint during the inference time(Hokamp and Liu, 2017; Post and Vilar, 2018; Susanto et al., 2020). These methods can typically satisfy most of the constraints but suffer from high computational cost and sometimes low translation quality because it always tries to strictly apply the terminology constraint regardless of the correctness of the whole sentence. The other group integrates lexical constraints during training (Dinu et al., 2019; Crego et al., 2016; Song et al., 2019) by annotating the data with special tags in order to guide the model to learn the enforcement of the translation constraints. The main disadvantage of these methods is the lack of guarantee of all constraints in the translations. Another limitation of these works is that they usually requires a term dictionary to augment the data, such extra resource is not always trivial to obtain for some domains in some languages.

Our work follows the second line of methods which incorporate terminology in the training by inserting special tags. Upon the recent works of (Dinu et al., 2019; Ailem et al., 2021), our system has made several improvements:

1. a terminology extracted from the given training corpus in a full non supervised manner rather than from a supervised approach or a given dictionary like in (Ailem et al., 2021).
2. only use special tags without source factors(Dinu et al., 2019) to annotate source and target terms in parallel sentences.
3. use a careful tag sentence re-sampling process to represent various constraint scenarios.

We evaluate our work on all the WMT23 terminology task including the blind test. Since the

Proceedings of the Eighth Conference on Machine Translation (WMT), pages 897–901
December 6–7, 2023. ©2023 Association for Computational Linguistics

reference is not released by the time of writing this paper, we evaluate our system by a simple naive strict match with respect to the target constraint. Our results show the effectiveness of the proposed method.

2 Method

In this section we present out system for the terminology task. Our approach is inspired by Ailem et al. (2021) and is further developed and adapted to this task.

2.1 Non supervised bilingual dictionary extraction

Approaches incorporating the constraints during the training time require some pre-built terminologies or dictionaries such as in the terminology task of *WMT2021*. The idea is to create training samples to guide the model to integrate the constraints when generating the output. However, in this year's shared task, terminology is not provided. Previous approaches such as Hazem and Morin (2016) and Liu et al. (2018) require heavy computation. Artetxe et al. (2016) can only learn single word bilingual lexicon. In this work, since our goal is to annotate the training data, having some noise in the extracted dictionary is affordable but the the number of the dictionary entries should be high enough to cover as much as possible different terminology constraint scenarios. Thereby we propose a simple yet efficient non supervised bilingual dictionary extraction approach which yields a large amount of aligned single and multi word items.

Our approach consists in extracting entries from two aspects: first we extract exact matching ngrams which contains more than 50% of non stop word or punctuation tokens from the two language texts, to prevent this process from being unnecessarily long, we limit the ngrams to five; the second aspect consists in extracting a whole sequence which is entirely included in another sequence of the corpus. The final dictionary contains both invariable and long sequence entries.

2.2 Data annotation

Following the work of Dinu et al. (2019) and Ailem et al. (2021), sentences matching source and target contraint terms are annotated with some special tags as illustrated in Figure 1.

Note that we also use mask tokens for the source term since this this provides a more general pattern for the model to learn to perform the copy operation every time it encounters the tag **<S>** followed by the **MASK** token. Moreover, this makes the model more apt to support conflicting constraints, i.e., constraints sharing the same source part but which have different target parts. This may be useful if some tokens must be translated into different targets for some specific documents and contexts at test time. Our preliminary experiments have shown the effectiveness of adding masks after data annotation.

Source	His critics state that this will just increase the **budgetary deficit** .
Constraint	**budgetary deficit** → **Haushaltsdefizit**
term annotation	His critics state that this will just increase the <S> **budgetary deficit** <C> **Haushaltsdefizit** </C> .
+MASK	His critics state that this will just increase the <S> **MASK MASK** <C> **Haushaltsdefizit** </C> .

Figure 1: Training data annotation.

2.3 Annotated data resampling

After the automatic data annotation, several filters are applied to construct a final tagged data set which equals to 20% of the original data. The goal is to cover different constraint contexts so that the model can learn all possible cases. The criterions of the filters are as follows:

Constraint length. Oversample constraints with more composing tokens.

Constraint occurrence. Oversample constraints with low occurrence.

Constraint number. Oversample sentences with different constraint numbers.

Constraint position. Make sure that constraints at the beginning, middle and end of a sentence follow a distribution of 10%, 80% and 10%.

For all the oversampling, we apply a modified version of the temperature sampling with a temperature equal to 5:

$$P_ts(t) = \frac{P(t)^{1/T}}{\sum_i P(t_i^{1/T})}$$

where $P_ts t$ is the temperature sampling probability for term t. T is the hyper-parameter temperature. $P(t)$ is the probability of term t, we assume it can be calculated by the following:

$$P_(t) = \frac{N(t)}{\sum_i N(t_i)}$$

where $N(t)$ is the frequency of term t in the training corpus. So $\sum_i N(t_i)$ represents actually the sum frequency of all terms. Finally the oversample size for term t, $N_{oversample}(t)$ will be the rounded up value of:

$$N_{oversample}(t) = P_t s(t) * \frac{N(t_{max})}{P_t s(t_{max})}$$

where t_{max} is the term having the highest frequency.

3 Experiments

3.1 Data

We participate in all three language pairs: Chinese to English (noted as zh2en), English to Czech (noted as en2cs) and German to English (noted as de2en). We use the corresponding parallel data provided by the general translation task and the development data of the terminology task. Since the given development set has only 100 sentences, we first oversample these 100 sentences by 10 times, then we randomly take 4000 sentences from given general data and add them to the oversampled data. This results in a final development set of 5000 sentences.

Regarding the training data annotation dictionaries, we extract invariable ngrams from one million random sentences. In addition, we follow what we have described in 2.1: sentences which are included in other longer sequences are added to the dictionary. An overview of the data is shown in 1.

Data	**size(sentence/item)**
zh2en train	33 892 215
zh2en dictionary	445 727
en2cs train	130 023 715
en2cs dictionary	559 063
de2en train	288 591 578
de2en dictionary	769 915

Table 1: Data used in the task

3.2 Settings

For all our translation models, we use a Transformer (Vaswani et al., 2017) with 6 stacked encoders/decoders and 8 attention heads as a building block for our systems. We also tie the source and target embeddings with the softmax layer with a shared source and target vocabulary. The model size is 512 for the source and target embeddings, 2048 for the inner layers of the fully connected feed-forward network and a dropout rate of 0.15.

Training batch size is set to 4000 tokens per iteration and we evaluate the model on the development set for every 5 000 iterations. The model is trained with an initial learning rate of 10^{-5} and 10 000 warm up steps. Training stop condition is 15 consecutive checkpoints without improvement. We use a length penalty of 0.65 and a beam size of 5 during inference for all models. All models are trained on two NVIDIA Geforce 2080Ti.

Before annotating the training data, we apply Moses tokenizer (Koehn et al., 2007)and we train a truecaser for each language and then truecase each language pair data. We also use *subword nmt*[1] to train a BPE (Sennrich et al., 2016) model of 50k merges.

3.3 Results

We evaluate our systems on the translation constraint success rate by a simple strict match because by the time of our naive evaluation, the reference was not available. We report our results on the test set in Table 2 with two settings: with and without terminology control.

	Accuracy% †	Accuracy% ‡
German to English	92.59	69.29
English to Czech	94.15	47.43
Chinese to English	83.77	22.21

†: with terminology, ‡: without terminology applied

Table 2: Term strict match accuracy (%) on the WMT23 testset with and without using extra terminology.

As shown in Table 2, our system achieves more than 90% accuracy on German to English and English to Czech test set. While the accuracy is obviously lower (roughly 10 points lower) on the Chinese to English test set, we think this might be related to the higher difficulty of the Chinese to English test set. In the test set, there are some

[1]https://github.com/rsennrich/subword-nmt

constraints which are basically named entity transcribed in *Pinyin*[2] script. For example, 段凌天 ⟶ *Duan Ling Tian* (Person), 武宗学府 ⟶ *the martial arts training institute* (Association). The model needs to somehow learn the transcription from Chinese character to *Pinyin* or a specific alignment on which there aren't much train data in the provided parallel corpus. As a whole, our system shows satisfying results when the terminology is provided. To study whether the high accuracy results are obtained by our terminology control system or not, we also evaluate our system but without giving any terminology during the inference. We should expect a big gap between the two settings (with and without terminology during the inference). The results confirm our assumption: an average of 40+ points of difference for the three directions.

For the blind test, we present our strict match accuracy in Table 3. The data in the blind test is provided in three different modes: the first one corresponds to general machine translation and the second one has the terminology dictionary added. The data is provided in three different modes. The last one has random, though correct, translations of words, which are not terminologies. The idea is to see if we obtain improvement between the different modes, in which case it means that the model is good at terminology control not because that it has learned the specific way of translating those terms but has learned how to make good use of terminology information.

	Accuracy% †	Accuracy% §	Accuracy% ‡
German to English	97.35	98.18	36.16
English to Czech	94.76	94.50	45.06
Chinese to English	93.26	74.20	48.45

†: with correct terminology; §: with random terminology; ‡: without terminology applied

Table 3: Term strict match accuracy (%) on the WMT23 blind testset with correct and random term, and without using extra terminology.

On all the language directions, our system achieve more than 90% accuracy when the terminology information is provided. When a random constraint is given, we consider the given random constraint term as the reference translation. In this case, we observe that our system can still obtain a high accuracy score. This means that the model is able to generalize the behavior of outputting any constraint. Finally, in the general translation setting, we see a sharp decreasing of the accuracy, 40+ points lower compared to the terminology setting. This phenomenon shows that the model is not just good on its own but can make good use of the terminology.

[2]en.wikipedia.org/wiki/Pinyin

4 conclusion

This paper describes our submission to the terminology shared task. We participate in three language directions, German to English, English to Czech and Chinese to English. We extract a bilingual dictionary for the three language directions in a fully non supervised way and train a neural machine translation model with augmented data for each direction. Our term strict match evaluation shows the effectiveness our proposed system for all the three directions.

5 Limitations

Since we pursue the line of works which incorporate terminology control by adding special tags during the training. This system has also the limit of not being able to guarantee the constraints to be present in the output because of the soft nature. This is mainly concertized by two cases:

- **No constraint**. The constraint is not presented at all in the translation.
- **Variant constraint**. The exact format of the constraint is not presented but a variant is proposed in the output.

We observe that for most of the time when the model fails to generate the target constraint, the scenario belongs to the second case which proposes a variant of the constraint. This translation is acceptable in a human evaluation context from time to time.

To address this main limit, we would like to exploit assembling our method with other techniques such as a post processing step to force the constraint if the constraint is not presented in the output.

References

Melissa Ailem, Jingshu Liu, and Raheel Qader. 2021. Encouraging neural machine translation to satisfy terminology constraints. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1450–1455, Online. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP'16)*, pages 2289–2294, Austin, TX, USA.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Josep Crego, Jungi Kim, Guillaume Klein, Anabel Rebollo, Kathy Yang, Jean Senellart, Egor Akhanov, Patrice Brunelle, Aurelien Coquard, Yongchao Deng, et al. 2016. Systran's pure neural machine translation systems. *arXiv preprint arXiv:1610.05540*.

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

Amir Hazem and Emmanuel Morin. 2016. Efficient data selection for bilingual terminology extraction from comparable corpora. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING'16)*, pages 3401–3411, Osaka, Japan.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, page 1535–1546.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Jingshu Liu, Emmanuel Morin, and Sebastián Peña Saldarriaga. 2018. Towards a unified framework for bilingual terminology extraction of single-word and multi-word terms. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING'18)*, pages 2855–2866, Santa Fe, NM, USA.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, page 1412–1421.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. *Proceedings of NAACL-HLT 2018*, page 1314–1324.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Kai Song, Yue Zhang, Heng Yu, Weihua Luo, Kun Wang, and Min Zhang. 2019. Code-switching for enhancing nmt with pre-specified translation. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, page 449–459.

Raymond Hendy Susanto, Shamil Chollampatt, and Liling Tan. 2020. Lexically constrained neural machine translation with levenshtein transformer. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 3536–3543.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.