Machine Translation that Peeks at the Reference

Vilém Zouhar
2023 project report ETH Zurich
vzouhar@inf.ethz.ch

Abstract

Machine translation with lexical constraints is a popular research topic, especially for terminology translation. Existing approaches for lexical control in MT are usually complex and not easily applicable to all existing MT toolkits. We propose an off-the-shelf baseline approach, **Peek MT**, for lexical constraints. During training, the model is provided with access to some of the words in the reference, allowing it to produce better translations. During inference, the user can specify which words they would like the translation to contain. Depending on the amount of additional tokens, the MT performance is improved by 1.3-4.4 BLEU points per revealed token. Despite these being very soft constraints, they are fulfilled ∼66% of the time. Notably, the same approach can also be used to control the output translation length without tinkering with the decoder. Finally, from analysis point of view, this method allows us to establish that the knowledge of particular word in the reference, such as verbs and organization names boosts the MT performance the most.

Code: `github.com/zouharvi/mt-peek`

1 Introduction

Controllable machine translation is gaining in popularity (Agrawal and Carpuat, 2019; Michon et al., 2020; Li et al., 2022) and especially controllable MT for terminology translation (Post and Vilar, 2018; Dinu et al., 2019; Exel et al., 2020). Approaches for lexical control in MT (i.e. *output has to contain the word X*), usually utilize methods in the decoder to satisfy these constraints. While successful, these approaches are also more complex and can not be used with all existing MT toolkits in case the specific functionality is not implemented there.

In this work, we aim to provide and analyze an extremely easy to use off-the-shelf baseline approach for lexical constraints. During training, the model has access to some of the words in reference and can use them to provide better translations (i.e. closer to the reference). During inference, the user can specify which words they desire to be in the translation using the same process. Note that no modification of the MT system is needed, apart from including additional tokens during training. Symbolically, model m takes in source sentence s and leaked information $\phi(r)$, which is based on the reference r. Formally, $m(s, \phi(r)) \xrightarrow[\text{REF}]{} r$.

Consider Example 1 where a user is translating the German sentence with the brand name *Zweifel*, which means, literally, *doubt*.[1] While the user may be unable to translate the sentence into English on their own, they are aware of the fact the output should contain the word *Zweifel* and so by including this in the model input, they are able to arrive at the correct translation.

$$m(\text{‘Ich esse gerne Zweifel-Chips’}, \emptyset)$$
$$\xrightarrow[\text{HYP}]{} \text{I like to eat doubt chips}$$
$$m(\text{‘Ich esse gerne Zweifel-Chips’}, \text{‘Zweifel’})$$
$$\xrightarrow[\text{HYP}]{} \text{I like to eat Zweifel chips}$$

Example 1: Incorrect and correct translation of a German sentence given additional info.

From an analysis perspective, by peeking into the reference, we are able to pinpoint which additional information is important for higher-quality translation. Inspired by this, we answer the following questions:

R1: How much does the reference help?
A1 (Section 3.1): Depending on the amount of information, up to 1.3-4.4 BLEU points per additional revealed and ordered token.

[1] de.wikipedia.org/wiki/Zweifel_(Unternehmen) As of early 2023, both Google Translate and DeepL provide the first, incorrect, translation.

R2: Which type of information from the reference is important?
A2 (Section 3.2): Including named entities, specifically organization names and verbs, adpositions and adjectives are the most efficient way of improving MT performance (2.4-4.0 additional BLEU points per each token).

R3: How hard are these trained soft constraints?
A3 (Sections 3.1 and 3.3): Controlling specific words depends on how related they are to the text and lexical constraints are fulfilled ∼66% of the time. Sentence length is controllable but degrades quality.

2 Experiment Setup

The model architecture is fixed to a single Transformer configuration from FairSeq (Ott et al., 2019). The training corpus consists of 10M sentence from CommonCrawl (El-Kishky et al., 2020) with the development and test corpus of 50k sentences each being sampled from the same distribution. The results presented in the main text of this paper are based only on German → English language direction and evaluated using BLEU. Note that this is a very restricted setup (see Limitations).

3 Results and Analysis

In this section, we will explore different parts of information form the reference which can be leaked into the model input. We use the term **per-token-utility** to denote how much one average leaked token contributes to the performance:

$$\textbf{per-token-utility} = \frac{\text{BLEU}_\text{x} - \text{BLEU}_\text{base}}{\text{avg. tokens}_\text{x}} \quad (1)$$

This allows us to compare the contribution of a specific leak type even when the amount of tokens is different.

3.1 Leaking Random Words

We first explore leaking random words from the reference to the MT system and observe the effect of how much of the reference is leaked. We do this by randomly sampling the reference and prepending the result to the MT input (see Example 2). The sampling output is fixed, so only a part of the reference is revealed to the system thorough the training. For 0%, the MT works as any other MT. For 0%, the MT has to only reshuffle the words from the reference. For this reason, we distinguish between the additional information being already correctly ordered or not.

$\overrightarrow{\text{Ref}}$He certainly goes into the offices, but are the offices really the castle?

- **Fully random leak**:

$\overrightarrow{\text{Src}}$ really certainly into castle | Er geht sicherlich in die Büros, aber sind die Büros wirklich das Schloss?
$\overrightarrow{\text{Hyp}}$He certainly goes into offices, but are the offices really the castle?

- **Synonyms**:

$\overrightarrow{\text{Src}}$ truly surely to fortress | Er geht sicherlich in die Büros, aber sind die Büros wirklich das Schloss?
$\overrightarrow{\text{Hyp}}$He surely goes to offices, but are the offices truly the fortress?

- **Random words**:

$\overrightarrow{\text{Src}}$ effort paint harmony approach | Er geht sicherlich in die Büros, aber sind die Büros wirklich das Schloss?
$\overrightarrow{\text{Hyp}}$He will surely approach the offices, but are the offices really paint?

Example 2: Effect of adversarial "leaked information."

The results show, that for both of these modes, the effect of additional information amount is *hyperlinear*. That is, the performance gains get larger when more words are already accessed. For example, the difference between 0% and 10% for the unordered model is 0.69 BLEU score and the difference between 40% and 50% of the same model is 2.70 BLEU score. The disprepancy between the two modes (ordered/unordered) from 50% onwards shows the lacking capability of the model to perform this reordering. Even when given the full reference, only reshuffled, the model achieves only ∼60 BLEU score.

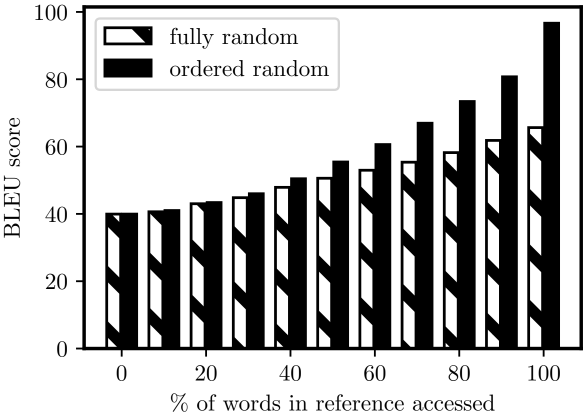

Figure 1: Model performance with a portion (%) of random words leaked. *Fully random* shuffles the word order while *ordered random* preserves it. See Example 2 for an illustration and Table 5 for tabular form.

The model is conditioned on input containing a given percentage of foreign words that always appear in the reference. However, there is no mechanism that would prevent this model from ignoring

Leaked type	% in hypothesis	BLEU
From reference	83%	45.30
Synonyms	48%	24.74
Random words	73%	25.56

Table 1: Proportion of words which appear in the model (30% of reference words) output with changing "leaked information."

the prefix with extra information. This conditioning creates a soft constraint and Table 1 shows how often it is satisfied. The results are mixed. When the word is present in the reference, it is satisfied 83% of the time. When the words are completely random and unrelated, the satisfaction is a bit lower, 73%. However, it is much more lower for synonyms of the words from the reference (using WordNet (Miller, 1998)). Furthermore, the lower performance is comparable for the random words and synonyms. An example of random words and synonyms is shown in Example 2. Nevertheless, the overall constraint satisfaction seems to be over 50%.

The inclusion of leaked information (30%) increases the probability of the specific tokens in the decoder output from 0.838 to 0.841 and of all others from 0.835 to 0.837. Even though the additional information improves the performance, it is not substantially reflected on the decoder confidence scores.

3.2 Leaking Words by POS and Entities

In this section we examine which words in particular are most helpful to the MT system. We distinguish between different types of words based on their part-of-speech (POS) and their entity types from named entity recognition. Because the distribution of POS in text is not uniform, we turn to per-token-utility to examine the usefulness of individual types. We note, that this quantity is far from perfect because of the observed hyperlinearity of performance contribution. Nevertheless, for word types which occur similar number of times, it provides a good comparison.

First, we consider words by their POS in Table 2. While leaking *nouns* leads to the highest performance overall, this effect is likely explained by nouns being the most common category. On the other hand, *verbs*, *adpositions* and *adjectives* occur 1-1.5× per segment and have per-token-utility of 2.4-3.7 BLEU scores (i.e. for every such revealed token, the BLEU score rises by this amount). Including all named entities (on average 1.1 per segment) has also a very positive impact (3.2 per-token-utility in Table 3). Most of it is due to including words which fall under the *organization* tag (4.0 per-token-utility). This confirms the intuition that named entities are the ones commonly mistranslated and giving the model some guidance improves the performance most efficiently (see Example 1).

POS	BLEU	Avg. tokens	Utility
-	39.94	0.0	-
Noun	53.66	4.73	2.90
Punct.	36.27	1.69	-2.17
Verb	45.29	1.46	3.66
Adp	42.96	1.22	2.48
Adj	42.31	0.99	2.40
Det	41.43	0.92	1.63
Pron	40.24	0.56	0.54
Num	40.67	0.54	1.35
Conj	40.05	0.52	0.21
Adv	40.76	0.37	2.20
Prt	40.21	0.26	1.04
Other	39.65	0.02	-17.20

Table 2: Effect of revealed words from the reference based on their POS.[2]

Entity Type	BLEU	Avg. tokens	Utility
-	39.94	0.0	-
All	43.54	1.14	3.15
Number	40.25	0.32	0.97
Organization	41.11	0.29	4.03
Name/event	39.59	0.23	-1.55
Date	40.32	0.17	2.28
Location	39.82	0.14	-0.85

Table 3: Effect of revealed words from the reference based on their named entity tag.[3]

3.3 Leaking Reference Length

Finally, another type of information that can be leaked from the reference is not specific words but rather the overall length. To this end, we consider

[2] We used NLTK (Bird et al., 2009) for POS tagging. Tagset: nltk.org/_modules/nltk/tag/mapping.html

[3] We used spacy (Honnibal and Montani, 2017) for named entity recognition.

two modes where we include the word count and subword count in the prefix. Example 3 shows, that by artificially changing this number, we gain some degree of control over the produced text at the cost of reduced quality of the translation. As documented in Table 4, including the additional tokens has a very small impact on the translation quality, but makes the translation lengths be much closer to the reference lengths.

$\overrightarrow{\text{SRC}}$ 4 | Du interpretierst alles falsch, sogar die Stille.
$\overrightarrow{\text{HYP}}$ You interpret everything wrong even silence

$\overrightarrow{\text{SRC}}$ 8 | Du interpretierst alles falsch, sogar die Stille.
$\overrightarrow{\text{HYP}}$ You interpret everything wrong, even the silence.

$\overrightarrow{\text{SRC}}$ 16 | Du interpretierst alles falsch, sogar die Stille.
$\overrightarrow{\text{HYP}}$ You are all interpreting wrong, even the silence of silence.

Example 3: Controlling length using prefix tokens.

4 Related Work

Terminology translation. Most MT systems for terminology use some form of contrained decoding (Hasler et al., 2018; Post and Vilar, 2018; Susanto et al., 2020). This has the disadvantages of added engineering and time complexity. In comparison to decoder-modification approaches to terminology, Dinu et al. (2019) train a blackbox MT, a similar setup to ours. Their approach is further similar by replacing specific words in the training data by the desired, already translated, terminology. However, we explore a different type of constraint, where we do not know which span in the source text the terminology is a translation for.

Access to reference. Li et al. (2022) use synthatic prompts during training to ellicit similar control in the model. However, the creation of these prompts is non-trivial and we additionally provide analysis of which word types specifically are important and how much these constraints are satisfied.

Leaked info.	BLEU	Length MAE Word	Length MAE Subword
-	39.94	1.39	2.57
Word Count	39.32	1.12	1.16
Subword count	39.13	1.05	0.53

Table 4: Effect of leaking reference sentence length (word or subword count) on model performance. The MAE shows mean average error between hypothesis and reference unit counts.

Infilling for MT can be seen as a variation on having access to the reference apart from the masked part and has been recently used in interactive scenarios (Xiao et al., 2022; Moslem et al., 2022).

Direct access to the reference was already explored from the perspective of a communication channel, i.e. leaked message has length constraints (Pal and Heafield, 2022a,b). However, these works use non-interpretable leakage of information (numbers and vectors), while our work focuses on leaking particular words. Another advantage of our approach is its potential use in terminology and interactive translation.

Output length control. Controlling the MT output length has first been explored by Lakew et al. (2019), who, among other approaches, also utilized similar approach to conditioning the model on length control tokens. They used a cruder approach to us, distinguishing only between short, normal and long ratios against the source.

5 Summary

In this work we presented an easy-to-use baseline MT with lexical and length constraints based around training with peeking at the reference.

- The additional information helps hyperlinearly (i.e. the more words already revealed the more they help per token).
- Verbs, adpositions, adjectives and organization names seem to be the most important for MT.
- Despite constraints being soft, they are more often than not satisfied, which justifies their use in production settings.
- Similar approach was demonstrated to work also with control over the sentence length.

6 Future work

- Examine peeking at the reference also in other generative NLP tasks.
- Leaking different types of information for different types of control to resolve ambiguity that arises during translation. For example, the formality which is not marked in English but is marked in German.[4]
- User study evaluating the efficacy of lexical constraints across translators working with given system, who are at different level of expertise in the target language.

[4] en.wikipedia.org/wiki/German_pronouns

Limitations

The provided experiments are only exploratory in nature and therefore have a limited setup, such as only one language direction (German → English) and one evaluation sentence-level metric (BLEU). Therefore, the results should not be interpreted as conclusive scientific results.

References

Sweta Agrawal and Marine Carpuat. 2019. Controlling text complexity in neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1549–1564.

Steven Bird, Ewan Klein, and Edward Loper. 2009. Natural language processing with python.

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. Training neural machine translation to apply terminology constraints. pages 3063–3068.

Ahmed El-Kishky, Vishrav Chaudhary, Francisco Guzmán, and Philipp Koehn. 2020. CCAligned: A massive collection of cross-lingual web-document pairs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 5960–5969, Online. 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.

Eva Hasler, Adrià de Gispert, Gonzalo Iglesias, and Bill Byrne. 2018. Neural machine translation decoding with terminology constraints. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 506–512.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Surafel Melaku Lakew, Mattia Di Gangi, and Marcello Federico. 2019. Controlling the output length of neural machine translation. In *Proceedings of the 16th International Conference on Spoken Language Translation*.

Yafu Li, Yongjing Yin, Jing Li, and Yue Zhang. 2022. Prompt-driven neural machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2579–2590.

Elise Michon, Josep M Crego, and Jean Senellart. 2020. Integrating domain terminology into neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3925–3937.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Yasmin Moslem, Rejwanul Haque, and Andy Way. 2022. Translation word-level auto-completion: What can we achieve out of the box? *arXiv preprint arXiv:2210.12802*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. FairSeq: A fast, extensible toolkit for sequence modeling. *NAACL HLT 2019*, page 48.

Proyag Pal and Kenneth Heafield. 2022a. Cheat codes to quantify missing source information in neural machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2472–2477, Seattle, United States. Association for Computational Linguistics.

Proyag Pal and Kenneth Heafield. 2022b. Cheating to identify hard problems for neural machine translation.

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.

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

Yanling Xiao, Lemao Liu, Guoping Huang, Qu Cui, Shujian Huang, Shuming Shi, and Jiajun Chen. 2022. BiTIIMT: A bilingual text-infilling method for interactive machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1958–1969, Dublin, Ireland. Association for Computational Linguistics.

		Fully random		Ordered random	
Revealed	**Avg. tokens**	**BLEU**	**Utility**	**BLEU**	**Utility**
0%	0.00	39.94	-	39.94	-
10%	0.84	40.63	0.82	41.00	1.25
20%	2.19	43.02	1.41	43.37	1.57
30%	3.43	44.82	1.42	45.99	1.76
40%	4.80	47.89	1.66	50.44	2.19
50%	6.25	50.59	1.70	55.42	2.48
60%	7.40	52.97	1.76	60.60	2.79
70%	8.64	55.36	1.78	66.94	3.12
80%	10.01	58.21	1.83	73.39	3.34
90%	11.23	61.83	1.95	80.73	3.63
100%	13.00	65.64	1.98	96.65	4.36

Table 5: Effect of revealed random words (with and without preserved ordering) from the reference.