# Negative Lexical Constraints in Neural Machine Translation

**Josef Jon** jon@ufal.mff.cuni.cz
**Dušan Variš** varis@ufal.mff.cuni.cz
**Michal Novák** mnovak@ufal.mff.cuni.cz
**João Paulo Aires** aires@ufal.mff.cuni.cz
**Ondřej Bojar** bojar@ufal.mff.cuni.cz
Institute of Formal and Applied Linguistics, Faculty of Mathematics and Physics, Prague, Czech Republic

**Abstract**

This paper explores negative lexical constraining in English to Czech neural machine translation. Negative lexical constraining is used to prohibit certain words or expressions in the translation produced by the neural translation model. We compared various methods based on modifying either the decoding process or the training data. The comparison was performed on two tasks: paraphrasing and feedback-based translation refinement. We also studied to which extent these methods "evade" the constraints presented to the model (usually in the dictionary form) by generating a different surface form of a given constraint.We propose a way to mitigate the issue through training with stemmed negative constraints to counter the model's ability to induce a variety of the surface forms of a word that can result in bypassing the constraint. We demonstrate that our method improves the constraining, although the problem still persists in many cases.

## 1 Introduction

In general, lexically constrained neural machine translation (NMT) is a method that allows enforcing presence or absence of certain words or phrases in the translation output . Positively constrained translation is more common and is used, for example, in named entities translation (Li et al., 2019; Yan et al., 2019), terminology integration (Dinu et al., 2019; Jon et al., 2021), or interactive machine translation (Knowles and Koehn, 2016).

Negative constraining serves different purposes. In this paper, we focus on two use-cases: (1) paraphrase generation and (2) refining translation based on feedback. Paraphrasing aims to produce a new translation hypothesis that differs from the original translation without significant changes in meaning. On the other hand, translation refinement involves replacing specific tokens in the original translation. These tokens can be selected either manually by the user or automatically using techniques like word-level quality estimation (Kepler et al., 2019). Negative constraining is particularly well-suited for translation refinement, while it can be one of the solutions for paraphrase generation.

After providing a summary of related work (Section 2), we proceed to describe the two tasks in detail (Section 3). Next, we delve into the methods we employ to achieve negative constraining (Section 4). The results are presented in Section 5, followed by a manual analysis of the outputs in Section 6.



*Proceedings of Machine Translation Summit XIX, Vol. 1: Research Track*, pages 372–384
September 4–8, 2023, Macau SAR, China.

## 2 Related work

There are three dominant approaches to constrained NMT. The earliest ones were based on replacing the constrained expressions in the source sentence with placeholders, ensuring that the placeholders are copied into the translation produced by the model and, finally, replacing the placeholders in the target with the desired expression (Crego et al., 2016; Hanneman and Dinu, 2020).

The second class of methods is based on modifying the decoding mechanism in such way that only translations including (or not including) the specified words or phrases can be produced in the final output (Anderson et al., 2017; Hasler et al., 2018; Chatterjee et al., 2017; Hokamp and Liu, 2017; Post and Vilar, 2018; Hu et al., 2019a).

The third class of methods revolves around altering the source input in the training data, allowing the NMT model to learn how to incorporate the constraints. This is typically done by either appending the constraints to the end of the source sentence as a suffix or intertwining them with the source sentence and distinguishing them from its tokens using factors (Dinu et al., 2019; Song et al., 2019; Chen et al., 2020; Jon et al., 2021; Bergmanis and Pinnis, 2021b,a).

Currently, most of the research in the field focuses on positive lexical constraints, often used for terminology integration. In contrast, there is a relatively less emphasis on negative constraining, despite its applications in areas like paraphrase generation (Hu et al., 2019b; Kajiwara, 2019). These works apply a method developed by Post and Vilar (2018) and later improved by Hu et al. (2019a). This method modifies the beam search decoding algorithm so that the beam in each time step includes the best hypotheses that satisfy from zero to the full number of pre-defined constraints. When using only negative constraints, the algorithm effectively boils down to filtering out hypotheses that would introduce any word (or phrase) from the list of constraints.

## 3 Task description

We carry out experiments with negative constraints in the two following tasks:

**Paraphrase generation** is often achieved through translation, where negative constraints come in handy for indicating the desired differences in the paraphrased output. To create a paraphrase of a source sentence, we go through multiple rounds of translation, each time disallowing some of the words generated in the previous pass. These restricted words or expressions should be replaced by synonymous expressions by the MT model, thereby creating a paraphrase of the original translation. As an example, consider the sentence *"He dodged the ball."* as the initial translation from a foreign language into English. When the word *"dodge"* is employed as the negative constraint, the system is expected to generate a paraphrase of the original translation (e.g. *"He avoided the ball."*) in the second pass.

**Feedback-based translation refinement** involves using external feedback to assess the model's output, for example, through user feedback in an interactive setting. After the initial translation is presented, the user can identify certain words as mistranslated. These words are then excluded from the subsequent output, prompting the model to generate a potentially improved translation. As obtaining human constraints can be costly, we translate the source without any constraints and analyze the tokens present in the MT output but not in the reference. In the next translation pass, we constrain the model to avoid using these "unconfirmed" tokens and evaluate the resulting translation.

In practice, word-level quality estimation (QE) systems can partially replace user feedback by highlighting potentially problematic tokens. In our work, we use references as a proxy for an oracle QE.

## 4 Proposed methods

We define a constraint as a sequence of consecutive subwords, which may represent either a single word or a multi-word expression. Each input example can have a list of multiple constraints that need to be satisfied. To incorporate these constraints into the translation process, we implement the following methods.

**Beam filtering** This method is based on an existing implementation where a hypothesis containing any forbidden subword is dropped from the beam search.[1] For each input sentence, a list of constraints (where each constraint represents a single subword) is provided. During beam search, any time a hypothesis that contain a constraint from the list is generated, it is removed. Optionally, it is removed only if the log probability of the subword is falls below a specified threshold. This method is referred to as the "subword method", and we extend it to support multi-subword expressions ("multi-subword method"). Instead of filtering after a single subword is generated, we store subwords corresponding to each constraint in a list of lists. For example:

- **Constraint 1:** decoding **Segmentation:** _deco ding
- **Constraint 2:** beam search **Segmentation:** _be am _search
- **Subword method:** [_deco, ding, _be, am, _search]
- **Multi-subword method:** [[_deco, ding], [_be, am, _search]]

Each hypothesis tracks its progress through the constraints, and it is removed only when a complete constraint is met. In other words, the hypothesis is removed only when all the subwords forming a single constraint are generated subsequently.[2]

**Score penalty** Another technique we experimented with is modifying the output probability of the subwords that form the constrained expression during the decoding. For this technique, we provided a list of constraints along with each input sentence. We created a mask with a penalty value for each subword present in the vocabulary. In our implementation, the penalty value was global, meaning each subword had either no or the same specified penalty. This mask was then summed with the output logits at each decoding step. To handle multi-subword constraints, we used a trie structure to track the progress through each constraint in each beam, similar to the approach used in (Hu et al., 2019a).

In the trie structure, each node represents a subword that is part of a constraint. The node contains a list of vocabulary IDs that, if generated in the next decoding step, would complete the constraint. When the subword represented by a node is produced, the penalty is added to the scores of these IDs in the next step.

**Learned constraints** A different approach to constraining involves modifying the training data to bias the model. The objective is to prevent the model from producing the constraint expressions that are directly provided with the input sentence. In our experiments, we separate the list of constraints from the source sentence by a special <sep> token, whereas the individual constraints within the list are separated by a special <c> token. For example:

- This is a sentence where we want to use synonyms for dog and cat. <sep> dog <c> cat

We train a model on the original dataset and the use this model to translate the source side of the dataset. Tokens present in the translation but not in the reference are extracted and used as "synthetic" constraints for training data, similar to the approach in the *Translation refinement* task. The resulting training dataset with "synthetic" constraints is then utilized to train a model capable of handling negative constraints in its input.

[1] Implemented here: https://github.com/XapaJIaMnu/marian-dev/tree/paraphrases_v2

[2] Link to the github repository of our code, removed for review.

| constraints | WMT20 | | Multi-ref | |
|---|---|---|---|---|
| | BLEU | COMET | BLEU | COMET |
| Yes | 30.8 | 0.6067 | 46.5 | 0.5971 |
| No | 30.7 | 0.6071 | 46.7 | 0.5944 |

Table 1: Comparison of the baseline models trained with and without constraints present in the training data. No constraints were present in the test set, showing that even the model exposed to the input constraints can be used in a "default" mode (no input constraints).

## 5 Experiments

In this section, we compare the performance of the methods on the tasks presented earlier.

### 5.1 Datasets and tools

We use CzEng 2.0 (Kocmi et al., 2020) dataset, all the authentic parallel sentences (61M), as the training dataset. We use WMT `newstest-2019` (Barrault et al., 2019) and `newstest-2020` (Barrault et al., 2020) for development and final evaluation respectively. We also used a subset of 50 examples from English-Czech `newstest-2011` which contains a large number of references (about 15M reference sentences in total, averaging 300k references per source sentence) introduced by Bojar et al. (2013) for part of the experiments. For evaluation on this multi-reference dataset (denoted "Multi-ref" in the following), we randomly picked up to 1,000 references for each source sentence to compute BLEU score and 20 references to compute COMET (the COMET scores are computed separately for each reference and averaged).

We use SentencePiece (Kudo and Richardson, 2018) for subword segmentation and UDPipe (Straka and Straková, 2017) for lemmatization. The models are trained with Marian (Junczys-Dowmunt et al., 2018) using default hyperparameters for Transformer-base architecture. BLEU (Papineni et al., 2002) scores are obtained by SacreBLEU (Post, 2018).[3] For COMET (Rei et al., 2020) scores, we evaluate with the *wmt20-comet-da* model. As the references in the Multi-ref test set are tokenized, we detokenized them using Sacremoses.[4]

### 5.2 Baseline

Our baseline model is a Transformer-base trained on CzEng 2.0 with negative constraints. This model is specifically trained to use negative constraints provided as part of the input, as described earlier in the *Learned constraints* section of Section 4. This approach enables more accurate comparison with other methods of incorporating constraints. Table 1 illustrates that when no constraints are provided at test time, the translation quality in terms of automated metrics is similar to a vanilla model without constraints.

### 5.3 Paraphrasing

In this task, our goal is to produce paraphrases that are diverse enough from the original translation. We thus opt for a multi-reference evaluation.

We create negative constraints by translating the source sentences of Multi-ref with the baseline model. The translations are then tokenized, removing punctuation and common Czech stopwords[5]. The remaining set of tokens serve as negative constraints.

[3]SacreBLEU signature: BLEU+case.mixed+lang.en-cs+numrefs.1+smooth.exp+test.wmt20+tok.13a+version.1.4.14

[4]`https://github.com/alvations/sacremoses`

[5]Prohibiting them by a constraint would hinder generation of grammaically fluent sentences.

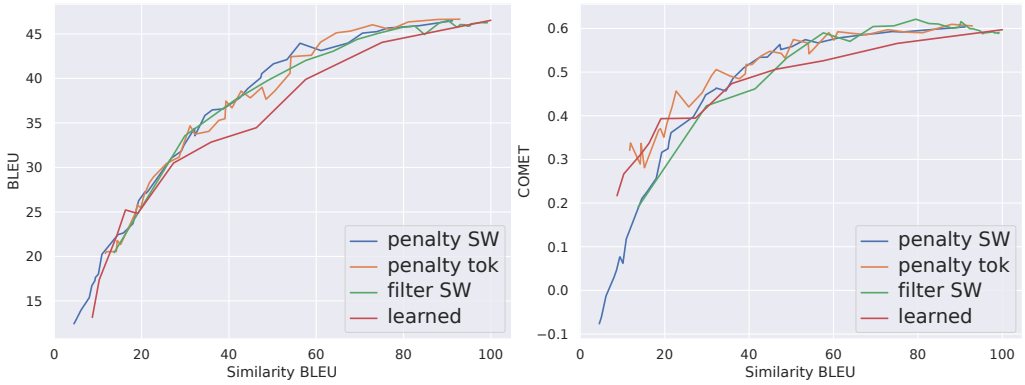


Figure 1: Correlation between either BLEU (left) or COMET (right) scores and similarity of translation to the baseline translation for paraphrasing.

| | Single subword | | | | Whole token | | | |
|---|---|---|---|---|---|---|---|---|
| **Penalty** | **↑BLEU** | **↓Sim** | **↑COMET** | **↓Cvg** | **↑BLEU** | **↓Sim** | **↑COMET** | **↓Cvg** |
| 0 | 46.5 | 100 | 0.5991 | 1.00 | 46.5 | 100 | 0.5991 | 1.00 |
| 0.1 | **46.5** | **83.6** | **0.5999** | 0.84 | **46.7** | **92.9** | **0.6078** | 0.94 |
| 0.2 | **45.9** | **76.4** | **0.5946** | 0.76 | **46.6** | **88.0** | **0.6123** | 0.89 |
| 0.5 | 45.1 | 70.6 | 0.5917 | 0.70 | **46.0** | **72.9** | **0.5991** | 0.73 |
| 1 | 41.6 | 50.2 | 0.5616 | 0.52 | 42.6 | 58.9 | 0.5939 | 0.62 |
| 2 | 32.5 | 29.7 | 0.4469 | 0.32 | 35.5 | 39.1 | 0.4988 | 0.46 |
| 3 | 20.2 | 10.9 | 0.1203 | 0.18 | 26.8 | 20.5 | 0.3869 | 0.30 |

Table 2: Results of the *score penalty* method on the paraphrasing task. We boldface variants where we deem the degradation small enough (BLEU or COMET close enough to their baseline value or even better).

In this task, our focus is on examining the relationship between the reference-based translation quality metrics (BLEU and COMET) and the similarity of the translation with the baseline translation. The objective is to generate sentences that are as distinct as possible while minimizing the negative impact on translation quality. The correlation for all the methods is depicted in Figure 1. Sampling across a range of thresholds (see below) generates various output variants. We arrange them on the x-axis based on their similarity with the unconstrained translation ("Similarity BLEU"). The y-axis then represents the automatically assessed translation quality. The curves' concave shape confirms that there is no sudden drop in quality as we paraphrase. However, even with the very permissive scoring against the Multi-ref references, both BLEU and COMET inevitably decline as we deviate further from the initial translations.

Tables 2–4 present the translation scores as well as the similarity of the paraphrase to the first translation (Similarity BLEU, denoted "Sim" here) for several thresholds. Each threshold controls the number of tokens to be paraphrased, affecting the similarity. However, its exact meaning differs for each method, as explained below. Coverage ("Cvg") indicates the ratio of constraint tokens that were produced in the translation (ignoring the casing).

The results for the *score penalty* method are presented in Table 2. *Penalty* represents the log probability that is subtracted from the logits for constrained tokens in each decoding step. Two variants of the method are compared. *Single subword* is the simpler variant, penalizing

| | Single subword | | | | Whole token | | | |
|---|---|---|---|---|---|---|---|---|
| **Thrshld** | **↑BLEU** | **↓Sim** | **↑COMET** | **↓Cvg** | **↑BLEU** | **↓Sim** | **↑COMET** | **↓Cvg** |
| 0 | 7.2 | 2 | -0.3388 | 0.07 | 8.7 | 2.8 | 0.0621 | 0.09 |
| -0.1 | 20.4 | 13.7 | 0.1919 | 0.17 | 18.1 | 10.5 | 0.2448 | 0.14 |
| -0.2 | 33.5 | 29.9 | 0.4285 | 0.37 | 33.9 | 26.8 | 0.4595 | 0.31 |
| -0.5 | 42.0 | 57.6 | 0.5938 | 0.60 | 41.7 | 53.3 | 0.5544 | 0.52 |
| -1 | **45.9** | **82.6** | **0.6146** | 0.83 | 45.1 | **77.8** | **0.6059** | 0.76 |
| -1.5 | **45.7** | **92.3** | **0.6011** | 0.91 | **46.1** | **89.8** | **0.6076** | 0.87 |
| -2 | **46.2** | **95.5** | **0.5901** | 0.96 | **46.2** | **93.3** | **0.5774** | 0.93 |
| -3 | **46.3** | **99.2** | **0.5931** | 0.99 | **46.3** | **99.1** | **0.5906** | 0.99 |

Table 3: Results of the *beam filtering* method on the paraphrasing task. Boldfacing as in Table 2.

| **Ratio** | **BLEU** | **Sim** | **COMET** | **Cvg** |
|---|---|---|---|---|
| 0 | 46.5 | 100 | 0.5991 | 1.00 |
| single | 45.4 | 81.4 | 0.5582 | 0.83 |
| 0.1 | 44.1 | 75.1 | 0.5685 | 0.76 |
| 0.2 | 39.9 | 57.6 | 0.5287 | 0.63 |
| 0.4 | 32.8 | 35.9 | 0.4796 | 0.43 |
| 0.6 | 24.8 | 19.1 | 0.4034 | 0.25 |
| 0.8 | 22.3 | 14.3 | 0.3193 | 0.18 |
| 1 | 13.1 | 8.7 | 0.2194 | 0.12 |

Table 4: Results of the *learned* method on the paraphrasing task. We do not boldface any row because the BLEU and COMET scores immediately degrade.

each subword found among the constraints. On the other hand, in the *Whole token* variant, the multi-subword implementation is used. The penalty is applied only when a whole constraint is completed in the hypothesis (in our configuration, the whole constraint will always be a single word, due to the constraint generation algorithm). The *penalty* parameter allows us to control the resulting paraphrase similarity: the higher its value, the more disadvantaged are the constrained tokens during decoding. We observe no significant degradation of translation up until about 88 BLEU similarity (0.89 coverage). Even at 72.9 BLEU similarity (0.73 coverage), the degradation is minimal. Multi-subword implementation yields better results than the single-subword implementation, allowing us to reach slightly lower coverage with comparable degradation, and it even appears to improve the baseline metric levels (BLEU of 46.7 and COMET of 0.6123 instead of the baseline 46.5 and 0.5991, respectively).

For the *beam filtering* method, the results are presented in Table 3. The controlling parameter is a threshold log probability, removing the hypotheses that use the constraint with a probability below the threshold. Opposed to the previous method, the lower its value, the more permissive the algorithm is, keeping the hypotheses with less probable constraints in the beam search. Again, two variants (single- and multi-subword) are implemented. For similar paraphrases, there are no notable score differences. However, as translations become more dissimilar, the multi-subword implementation performs better. Overall, *beam filtering* and *score penalty* methods show similar performance. An improvement in overall quality in terms of COMET is again observed when deviating somewhat from the baseline output (COMET slightly above 0.60 compared to 0.59).

Results for the *learned* constraints method are displayed in Table 4. We consider content


Figure 2: The best results obtained by each method on the *translation refinement* task, either in terms of BLEU (left) or COMET (right) scores. These results were computed using the best found setting of the control parameter for each method.

| | Single subword | | | | Whole token | | | |
|---|---|---|---|---|---|---|---|---|
| **Penalty** | **BLEU** | **Sim** | **COMET** | **Cvg** | **BLEU** | **Sim** | **COMET** | **Cvg** |
| 0 | 46.5 | 100 | 0.5991 | 1 | 46.5 | 100 | 0.5991 | 1 |
| 0.1 | 46.9 | 95.4 | **0.6144** | 0.93 | 47.0 | 96.5 | 0.6104 | 0.95 |
| 0.5 | 48.5 | 80.6 | 0.6024 | 0.70 | 48.7 | 85.1 | 0.6237 | 0.76 |
| 1 | **48.6** | 68.9 | 0.5754 | 0.50 | 48.5 | 74.3 | **0.6302** | 0.59 |
| 2 | 47.1 | 57 | 0.5773 | 0.30 | 48.6 | 63.9 | 0.6011 | 0.43 |
| 3 | 48.2 | 53.3 | 0.5617 | 0.19 | 49.4 | 61.4 | 0.5790 | 0.33 |
| 3.5 | 48.1 | 50.8 | 0.5226 | 0.15 | **49.4** | 57.1 | 0.5695 | 0.22 |

Table 5: Results of the *score penalty* method on the refinement task.

words from the baseline translation as potential negative constraints, resulting in a full set of conceivable constraints for a sentence. The method's control parameter is the ratio of total constraints to those actually used. For example, with 6 available constraints for a sentence and a ratio of 0.5, we select only 3 constraints. "Singl" in the ratio column indicates that only one constraint was used for each sentence. The selection is based on token-level model scores from the baseline translation, where scores of subwords comprising a token are summed. The lowest log probability tokens are constrained first, effectively preventing the usage of words that the baseline model hesitates to produce. We chose this sampling approach after observing large result variances when using randomly sampled constraints. However, we acknowledge that this selection method is not optimal, as several random runs led to significantly better BLEU and COMET scores. The learned constraints underperform compared to other approaches, likely because the decoding-based methods offer more precise control over which constraints to use (penalty or threshold).

### 5.4 Translation refinement

Unlike the paraphrasing task, where the relationship between similarity and translation quality is relevant, the translation refinement task solely aims to improve the absolute quality of translation. The best scores achieved with optimal control parameters are presented in Figure 2.

Results for *score penalty* and *beam filtering* methods are presented in Tables 5 and 6, showing the similar performance to each other, as already observed in the previous task.

In the *learned constraints* method (Table 7), the BLEU scores improve with an increasing ratio of constraints, while the COMET scores do not follow the same trend.

The *learned constraints* method outperformed others significantly in terms of BLEU score. The *score penalty* method achieved a slightly better COMET score with the best penalty value.

| | Single subword | | | | Whole token | | | |
|---|---|---|---|---|---|---|---|---|
| **Thrshld** | **BLEU** | **Sim** | **COMET** | **Cvg** | **BLEU** | **Sim** | **COMET** | **Cvg** |
| 0 | 47.4 | 48.7 | 0.4755 | 0.03 | 49.4 | 50.1 | 0.5771 | 0.05 |
| -0.1 | 47.9 | 52.4 | 0.6012 | 0.19 | **49.6** | 53.4 | 0.5814 | 0.16 |
| -0.2 | 48.7 | 59.5 | 0.6163 | 0.37 | 48.7 | 56.7 | **0.6192** | 0.31 |
| -0.3 | **49.4** | 65.7 | 0.5976 | 0.46 | 48.5 | 63.7 | 0.6179 | 0.42 |
| -1 | 47.1 | 88 | **0.6100** | 0.83 | 47.7 | 85.4 | 0.6109 | 0.76 |
| -2 | 46.3 | 96.9 | 0.5932 | 0.97 | 46.5 | 95 | 0.5813 | 0.93 |
| -3.5 | 46.3 | 99.2 | 0.5931 | 0.99 | 46.3 | 99.2 | 0.5931 | 0.99 |

Table 6: Results of the *beam filtering* method on the refinement task.

| **ratio** | **BLEU** | **Sim** | **COMET** | **Cvg** |
|---|---|---|---|---|
| 0 | 46.5 | 100 | 0.5991 | 1.00 |
| single | 47.6 | 82.3 | 0.6123 | 0.75 |
| 0.1 | 46.8 | 94.4 | 0.6058 | 0.92 |
| 0.2 | 47.0 | 83 | **0.6212** | 0.75 |
| 0.4 | 47.4 | 72.5 | 0.6026 | 0.56 |
| 0.6 | 48.7 | 65.7 | 0.5922 | 0.38 |
| 0.8 | 51.2 | 58.8 | 0.6103 | 0.21 |
| 1 | **53.4** | 55.4 | 0.5746 | 0.08 |

Table 7: Results of the *learned* method on the refinement task.

We believe this is again due to the decoding methods providing more precise control over the enforcement of constraints compared to the learned method.

In Table 8 we present results for the two best scoring methods on a better-known test set for comparison, *newstest20* (Barrault et al., 2020). The *learned* method provides better results than the *score penalty* method on this dataset.

## 6 Manual analysis

Our results show that the methods tend to overlook some negative constraints and still produce prohibited words. Both the *score penalty* and *beam filtering* methods require pushing the thresholds quite far to satisfy all constraints. Conversely, the *learned* method is more attentive to constraining but results in quick degradation of translation quality. To gain insights into the system behavior, we examined the outputs and present typical examples for each class in Figure 3. These examples are from the *translation refinement* task using the *learned* method, with constraints being tokens present in the baseline translation but not in the reference. The first example showcases a clear failure of the method, as the constraint is ignored without any apparent reason. The second example is challenging, as it requires knowledge of the Czech transcription of the name *Assam* based on its English transcription.

The *Reference* error example illustrates a situation, where the the meaning of the reference translation that we use to generate the negative constraints slightly deviates from the source sentence, resulting in a constraint difficult to satisfy. The reference translation replaces the term *two-thirds* (*dvoutřetinovou*) with a different term, *needed* (*potřebnou*), which leads to *dvoutřetinovou* being selected as a constraint. Since it is difficult to translate *two-thirds majority* differently from the baseline translation, the model fails to do so. This issue could be addressed

| | Learned | | Score penalty | | |
|---|---|---|---|---|---|
| **ratio** | **BLEU** | **COMET** | **penalty** | **BLEU** | **COMET** |
| single | 31.5 | **0.6183** | 0.2 | 30.6 | **0.6033** |
| 1 | **38.5** | 0.5973 | 0.1 | **30.8** | 0.6028 |
| baseline | 30.9 | 0.6067 | | | |

Table 8: Results of best performing methods on newstest20. Results obtained using best-performing parameters for both metrics separately are shown.

| **Model** | **Constraints** | **BLEU** | **Surface Form Cvg** | **Lemma Cvg** |
|---|---|---|---|---|
| SF | no | 30.9 | 1.00 | 0.96 |
| SF | SF | 38.5 | 0.09 | 0.34 |
| Stem | no | 30.9 | 1.00 | 0.96 |
| Stem | Stem | 36.9 | 0.22 | 0.39 |

Table 9: Comparison of surface form and lemma coverage (Cvg) for models trained with either surface form or stemmed constraints. Evaluated on `newstest-2020`.

by using a validation dataset with more accurate reference translations.

In the *Segmentation* error example, the constraint is circumvented by employing a different subword segmentation of the output. Sinve we use SentencePiece without prior tokenization, adding a quotation mark („) at the beginning of a token results in a different segmentation that is not accounted for by the constraints (as the constraints are provided to the model with pre-existing segmentation).

The *Inflection* example demostrates a scenario where the model managed to avoid generating a constraint in a specific form but did not avoid producing the constrained term itself. Out of 8 constraints, 4 are fulfilled with a different inflected form in the constrained translation (in addition, one constraint is produced with a different spelling: *diskusi/diskuzi*). This behavior is undesirable because such circumvention can still lead to a potentially problematic translation. However, in certain cases, like paraphrasing, it may be deemed acceptable.

The extent of this behavior is presented in Table 9. We conduct a comparison between coverage at the surface form level and coverage at the lemma level. The evaluation is based on the *translation refinement* task on `newstest-2020`, using the *learned* method with a constraint usage ratio of 1.0. For the lemma-level coverage assessment, both the constraints and constrained translation were lemmatized. This ensures that even when the constraint is generated in a different surface form, it is considered covered. It is important to note that our lemmatization method is context-dependent, and in some cases, different lemmas may be produced for the same word in a sentence and in the constraint list, leading to some imprecision in these results.

At the surface level, the coverage is 0.09, indicating that 91% of the constraints are correctly satisfied. However, at the lemma level, the coverage increases to 0.34, which means that another 25% of the constraints appear in the translation in a different surface form, not detected by the previous method of computing coverage. We attempted to mitigate this behavior by training the model to use stemmed constraints (*Stem* model in Table 9). Our goal was to leverage the language modeling capability of the NMT model to account for all the possible word forms. While this approach partially works, reducing the gap between surface form and lemma coverage to 17 instead of 25, the overall performance is inferior (BLEU of 36.9 instead of 38.5).

| Source | Base translation | Constraints | Constrained translation | Error |
|---|---|---|---|---|
| Michael Jackson's former bodyguard has claimed the late singer cultivated some of his eccentricities with the deliberate intention of **riling up** the media. | Bývalý bodyguard Michaela Jacksona tvrdil, že zesnulý zpěvák pěstoval některé z jeho výstředností s úmyslem rozzuřit média. | bodyguard, tvrdil, pěstoval, své, výstřednosti, s, úmyslem, **rozzuřit**, média | Bývalý osobní strážce Michaela Jacksona tvrdí, že zesnulý zpěvák pěstuje některé z jeho výstředností se záměrem **rozzuřit** sdělovací prostředky. | Not satisfied |
| And Modi's government has created an uproar by instituting a national registry of citizens and setting up detention camps in the border state of Assam. | A Modiho vláda vyvolala pozdvižení zavedením národního registru občanů a zřízením zadržovacích táborů v pohraničním státě **Assam** | Modiho, vyvolala, pozdvižení, zavedením, zadržovacích, **Assam** | A Módího vláda způsobila rozruch vytvořením národního registru občanů a zřízením zajateckých táborů v pohraničním státě **Assam**. | Challenging |
| Neither chamber of Congress appears to have the two-thirds majority needed to override the president's opposition. | Zdá se, že ani jedna kongresová komora nemá **dvoutřetinovou** většinu potřebnou k překonání prezidentovy opozice. | kongresová, komora, **dvoutřetinovou**, překonání | Zdá se, že ani jedna z kongresových komor nemá **dvoutřetinovou** většinu potřebnou k potlačení prezidentovy opozice. | Reference |
| _Last _year , _construction _of _Q id di y a _" ent er tain ment _city " _was _launched _near _Ri y ad h. | _Po bl í ž _Ri já du _byla _v _loňském _roce _zahájen a _výstavba **_útvar ového** _města _Q id di y a | **_útvar ového** | Po bl í ž _Ri já du _byla _v _loňském _roce _zahájen a _výstavba _„ **ú t var ového** _města “ _Q id di y a | Segmentation |
| A Pittsburgh native whose real name was Malcolm James Myers McCormick, Miller's lyrics included frank discussion of his depression and drug use. | Domorodec z Pittsburghu, jehož pravé jméno bylo Malcolm James Myers McCormick, Millerovy texty zahrnovaly upřímnou diskusi o jeho depresi a užívání drog. | **domorodec**, pravé, **Millerovy**, **texty**, zahrnovaly, upřímnou, **diskusi**, **depresi** | **Domorodce** z Pittsburghu, jehož skutečné jméno bylo Malcolm James Myers McCormick, **Millerův text** obsahoval otevřenou **diskuzi** ohledně **deprese** a užívání drog. | Inflection |

Figure 3: Examples of baseline and constrained translations with interesting behavior. The columns show the English source sentence, baseline translation into Czech, list of constraints, and the final constrained translation. The last column contains a type of error observed. The Segmentation example is shown in subword units for explanation purposes.

## 7 Conclusion

We conducted a thorough investigation into NMT decoding with negative lexical constraints, addressing two tasks: paraphrasing and interactive translation refinement. Our comparison of various approaches revealed that it is indeed possible to restrict the NMT model from generating specific words in its output. However, none of the methods provided flawless results. By examining the errors made by the most effective approach, we identified instances where the model evades the constraints in morphologically rich languages by producing slightly different surface forms of the prohibited words. While we proposed a simple solution by training the model to use stemmed constraints, it adversely impacts the overall translation quality. Despite these challenges, our research sheds light on the potential of using negative constraints in NMT decoding and highlights areas for further improvement.

## Acknowledgements

This work was partially supported by the Charles University project GAUK No. 244523, the grant 825303 (Bergamot) of the European Union's Horizon 2020 research and innovation programme, the grant 19-26934X (NEUREM3) of the Czech Science Foundation and the grant FW03010656 of the Technology Agency of the Czech Republic.

## References


Anderson, P., Fernando, B., Johnson, M., and Gould, S. (2017). Guided open vocabulary image captioning with constrained beam search. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 936–945, Copenhagen, Denmark. Association for Computational Linguistics.

Barrault, L., Biesialska, M., Bojar, O., Costa-jussà, M. R., Federmann, C., Graham, Y., Grundkiewicz, R., Haddow, B., Huck, M., Joanis, E., Kocmi, T., Koehn, P., Lo, C.-k., Ljubešić, N., Monz, C., Morishita, M., Nagata, M., Nakazawa, T., Pal, S., Post, M., and Zampieri, M. (2020). Findings of the 2020 conference on machine translation (WMT20). In *Proceedings of the Fifth Conference on Machine Translation*, pages 1–55, Online. Association for Computational Linguistics.

Barrault, L., Bojar, O., Costa-jussà, M. R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Koehn, P., Malmasi, S., Monz, C., Müller, M., Pal, S., Post, M., and Zampieri, M. (2019). Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

Bergmanis, T. and Pinnis, M. (2021a). Dynamic terminology integration for COVID-19 and other emerging domains. In *Proceedings of the Sixth Conference on Machine Translation*, pages 821–827, Online. Association for Computational Linguistics.

Bergmanis, T. and Pinnis, M. (2021b). Facilitating terminology translation with target lemma annotations. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3105–3111, Online. Association for Computational Linguistics.

Bojar, O., Macháček, M., Tamchyna, A., and Zeman, D. (2013). Scratching the surface of possible translations. In Habernal, I. and Matoušek, V., editors, *Text, Speech, and Dialogue*, pages 465–474, Berlin, Heidelberg. Springer Berlin Heidelberg.

Chatterjee, R., Negri, M., Turchi, M., Federico, M., Specia, L., and Blain, F. (2017). Guiding neural machine translation decoding with external knowledge. In *Proceedings of the Second Conference on Machine Translation*, pages 157–168, Copenhagen, Denmark. Association for Computational Linguistics.

Chen, G., Chen, Y., Wang, Y., and Li, V. O. (2020). Lexical-constraint-aware neural machine translation via data augmentation. In Bessiere, C., editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3587–3593. International Joint Conferences on Artificial Intelligence Organization. Main track.

Crego, J., Kim, J., Klein, G., Rebollo, A., Yang, K., Senellart, J., Akhanov, E., Brunelle, P., Coquard, A., Deng, Y., Enoue, S., Geiss, C., Johanson, J., Khalsa, A., Khiari, R., Ko, B., Kobus, C., Lorieux, J., Martins, L., Nguyen, D.-C., Priori, A., Riccardi, T., Segal, N., Servan, C., Tiquet, C., Wang, B., Yang, J., Zhang, D., Zhou, J., and Zoldan, P. (2016). Systran's pure neural machine translation systems.

Dinu, G., Mathur, P., Federico, M., and Al-Onaizan, Y. (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.

Hanneman, G. and Dinu, G. (2020). How should markup tags be translated? In *Proceedings of the Fifth Conference on Machine Translation*, pages 1160–1173, Online. Association for Computational Linguistics.

Hasler, E., de Gispert, A., Iglesias, G., and Byrne, B. (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, New Orleans, Louisiana. Association for Computational Linguistics.

Hokamp, C. and Liu, Q. (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.

Hu, J. E., Khayrallah, H., Culkin, R., Xia, P., Chen, T., Post, M., and Van Durme, B. (2019a). Improved lexically constrained decoding for translation and monolingual rewriting. In *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)*, pages 839–850, Minneapolis, Minnesota. Association for Computational Linguistics.

Hu, J. E., Rudinger, R., Post, M., and Durme, B. V. (2019b). Parabank: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation.

Jon, J., Aires, J. P., Varis, D., and Bojar, O. (2021). End-to-end lexically constrained machine translation for morphologically rich languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4019–4033, Online. Association for Computational Linguistics.

Junczys-Dowmunt, M., Grundkiewicz, R., Dwojak, T., Hoang, H., Heafield, K., Neckermann, T., Seide, F., Germann, U., Aji, A. F., Bogoychev, N., Martins, A. F. T., and Birch, A. (2018). Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Kajiwara, T. (2019). Negative lexically constrained decoding for paraphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6047–6052, Florence, Italy. Association for Computational Linguistics.

Kepler, F., Trénous, J., Treviso, M., Vera, M., and Martins, A. F. T. (2019). OpenKiwi: An open source framework for quality estimation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 117–122, Florence, Italy. Association for Computational Linguistics.

Knowles, R. and Koehn, P. (2016). Neural interactive translation prediction. In *Conferences of the Association for Machine Translation in the Americas: MT Researchers' Track*, pages 107–120, Austin, TX, USA. The Association for Machine Translation in the Americas.

Kocmi, T., Popel, M., and Bojar, O. (2020). Announcing czeng 2.0 parallel corpus with over 2 gigawords. *CoRR*, abs/2007.03006.

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

Li, X., Yan, J., Zhang, J., and Zong, C. (2019). Neural name translation improves neural machine translation. In Chen, J. and Zhang, J., editors, *Machine Translation*, pages 93–100. Springer Singapore.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Post, M. (2018). A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Post, M. and Vilar, D. (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.

Rei, R., Stewart, C., Farinha, A. C., and Lavie, A. (2020). COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Song, K., Zhang, Y., Yu, H., Luo, W., Wang, K., and Zhang, M. (2019). Code-switching for enhancing NMT with pre-specified translation. In *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)*, pages 449–459, Minneapolis, Minnesota. Association for Computational Linguistics.

Straka, M. and Straková, J. (2017). Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.

Yan, J., Zhang, J., Xu, J., and Zong, C. (2019). The impact of named entity translation for neural machine translation. In Chen, J. and Zhang, J., editors, *Machine Translation*, pages 63–73. Springer Singapore.