Morphology-Inspired Word Segmentation for Neural Machine Translation

Jānis Zuters a, [[1]](#footnote-1), Gus Strazds a, [[2]](#footnote-2), and  
Viktorija Ļeonovaa

a University of Latvia, Raina blvd. 19, LV-1586 Riga, Latvia

**Abstract.** This paper proposes the Prefix-Root-Postfix-Encoding (PRPE) algorithm, which performs close-to-morphological segmentation of words as part of text pre-processing in machine translation. PRPE is a cross-language algorithm requiring only minor tweaking to adapt it for any particular language, a property which makes it potentially useful for morphologically rich languages with no morphological analysers available. As a key part of the proposed algorithm we introduce the ‘Root alignment’ principle to extract potential sub-words from a corpus, as well as a special technique for constructing words from potential sub-words. In addition, we supplemented the algorithm with a specific processing for named-entities based on transliteration. We conducted experiments with two different neural machine translation systems, training them on parallel corpora for English-Latvian and Latvian-English translation. Evaluation of translation quality showed improvements in BLEU scores when the data were pre-processed using the proposed algorithm, compared to a couple of baseline word segmentation algorithms. Although we were able to demonstrate improvements in both translation directions and for both NMT systems, they were relatively minor, and our experiments show that machine translation with inflected languages remains challenging, especially with translation direction towards a highly inflected language.

**Keywords:** Neural machine translation, Word segmentation, Named-entity processing.

# Introduction

In recent years neural machine translation (NMT) has indisputably become the default approach for machine translation. Still, the quality of translation differs widely depending on the language pairs involved – for morphologically rich languages, especially those with relatively small amounts of available parallel training data, training an NMT system remains challenging due to data sparseness [1].

To overcome data sparsity due to inflectedness of a language, it is common to apply various forms of data pre-processing, and one of the most commonly used techniques is splitting words into segments (or sub-words) in order to decrease the amount of unique input tokens. This reduces data sparseness to the extent that a large number of lexicographically unique word tokens (in morphologically rich languages, these include the many inflected forms of each individual word) can be represented as combinations built up from a much smaller vocabulary of sub-word tokens. This is important because of the main paradigm of NMT – sequence to sequence transduction of text units (characters, or words, or sub-words) that are seen and processed by the system as indivisible tokens. A good segmentation into sub-word tokens would have the property that specific word forms which were not present in the training data at all can nevertheless be represented as a sequence of tokens from the sub-word vocabulary, and, ideally, the neural network could learn to generate correct (but possibly not previously encountered) output sequences for previously unseen specific input sequences (e.g. producing word forms that are correctly inflected even though they might not have been present in the training datasets).

This article focuses on word segmentation implicitly based on sub-word statistics (Prefix-Root-Postfix-Encoding algorithm, PRPE). The output text resembles morphologically segmented text, but without making any claims to being a linguistically well-motivated morphological splitting. Thus, the output of the proposed segmentation method was not compared against a reference segmentation (as is done, for example, in [2]). Producing reference segmentations for a large corpus of text requires considerable effort (and, usually, some amount of non-trivial linguistic theory concerning the morphological structure of the specific language begin analysed). Instead, experiments were conducted to directly test whether PRPE segmentation improves translation quality relative to a couple of baseline segmentation schemes. Unlike language specific morphological segmenters, PRPE is almost language independent, requiring relatively little work (a handful of new or modified lines of code and some parameter tuning) to adapt it to a new language.

# Related Work

This paper focuses on a particular approach of text pre-processing for NMT to overcome inflectedness of languages and the resultant problem of sparsity of specific word forms in training corpora – segmentation of text into sub-word units. This section gives a brief overview of some commonly used sub-word segmentation algorithms.

## Byte Pair Encoding Based Segmentation Algorithm

Byte pair encoding based segmentation algorithm (BPE), proposed in [3], utilizes the principle of iteratively finding the most frequent character sequences of the text to become potential segments (see a segmentation example in Table 1).

The algorithm consists of two phases: (a) the learning phase, in which the vocabulary of merge operations is obtained, (b) the apply phase, in which a specific text is segmented using the vocabulary.

The learning phase starts with all words in the text represented as sequences of characters. Then, through an iterative process, the most frequent pairs of neighbouring symbols (initially, characters) are merged together and these pairs (or ‘merge operations’) are written to a special vocabulary. At each iteration, (a) the chosen merge operation is added to the vocabulary, (b) the merge operation is applied to the text. The process is continued until a predefined number of merge operations is reached.

The apply phase transforms input text into segmented text according to the vocabulary of merge operations.

BPE provides control over the effective size of the vocabulary for translation, since the vocabulary of unique tokens after applying BPE is less than or equal to the number of unique characters in the original input text plus the number of merge operations. A bounded vocabulary is essential for typical approaches to NMT, and since the introduction of the BPE algorithm adapted for this purpose in [3], it has become something of a standard practice to pre-process input text for NMT by segmenting it with BPE.

**Table 1.** A segmentation example with BPE.

|  |  |
| --- | --- |
| **Language** | **Segmented Text** |
| English | you need to know exactly what you want to im–mor–tal–ise during the photo session , and be able to tell the photo–grap–her about it . |
| Latvian | ir jāsaprot , ko tieši tu vē–lies ie–mūž–ināt foto–sesijas laikā un jāpa–stāsta par to fotogrāf–am . |

## Morphology-Driven Splitting

One of ideas for word segmentation for NMT is trying to separate roots from affixes (especially suffixes in morphologically rich languages), in the hope that doing so will preserve more semantic information (words with common roots would also have the same segments).

In [1] a language-specific morphological splitting approach is described (see a segmentation example in Table 2). To avoid over-segmentation of the text, morphological splitting is performed in a limited manner, i.e., not all affixes are separated (too many segments in a sequence reduces the quality of NMT).

For translation between English and Latvian, morphology-driven splitting was found to give a small improvement on translation quality (0.5-0.7 BLEU points, [4]) relative to BPE. The small improvement might be explained by a relatively small out-of-vocabulary rate given the training data used (especially in English).

**Table 2.** A segmentation example with morphology-driven splitting proposed by [1] (postprocessed with BPE to support open vocabulary).

|  |  |
| --- | --- |
| **Language** | **Segmented Text** |
| English | you need to know exact–ly what you want to im–mor–tal–ise during the photo session , and be able to tell the photo–grap–her about it . |
| Latvian | ir jā–saprot , ko tieš–i tu vēl–ies ie–mūž–inā–t foto–sesij–as laik–ā un jā–pastāst–a par to foto–grāf–am . |

Morphology-driven splitting is typically carried out using language-specific morphological analysers. Building such analysers for inflective and agglutinative languages is more complicated than for English (see [5]). For example, for many languages morphological analysis must deal with a considerable amount of ambiguity, and therefore various disambiguation models are used ([6], [7]). As morphological analysers are typically language specific, it takes a lot of effort to build such a tool for any given language (e.g. creating morphologically annotated corpora, developing language specific routines).

## Named-entity Processing for Machine Translation

In a machine translation task, translation of rare words, e.g. named-entities, is a challenging issue. Paper [1] proposes to modify splitting algorithm by keeping together unknown parts of a word (i.e., without splitting them) to be able then to process them differently.

One of the approaches for translation of named-entities is transliteration. In [trans2017], a successful use of neural networks for transliteration is described.

# PRPE Segmentation Algorithm

## General Description

This section describes the basic principles of the proposed Prefix-Root-Postfix-Encoding (PRPE) algorithm[[3]](#footnote-3) (see a segmentation example in Table 4) as well as a prototype of specific processing named entities accompanying the algorithm[[4]](#footnote-4). The main motivation for the algorithm is the belief that splitting away roots from words would produce more meaningful parallel sequences for machine translation (as with morphology-driven splitting, see section 2.2), thus increasing the quality of machine translation. But the goal of PRPE is to obtain such a segmentation based primarily on the statistics of the training data, using a bare minimum of language specific knowledge (contrast this with a language-specific morphological analyser, which would be hand-crafted using a large number of language-specific rules based on a linguistically motivated analysis of the morphological processes at work in a given language).

The basic principle underlying PRPE comes from the BPE algorithm – to learn the most frequent character sequences and then use them to segment words in a text. The main idea added is to take the most frequent left and right substrings of words instead of any character sequences, regarding left substrings as potential prefixes and roots, but right substrings as potential postfixes. Then these potential building blocks (prefixes, roots, postfixes) are combined together in a special way to constitute words – thus performing segmentation. As a result, a close-to-morphological segmentation is obtained. For better results, the PRPE algorithm should be complemented with a small language specific part. Instead of complicated probability computations, in PRPE we use substring frequencies and lists of substrings specifically ranked according to frequencies.

**Table 4.** A segmentation example with PRPE (post-processed with BPE to support open vocabulary). Linguistically, morphological splitting is similar in Latvian and English. The two main differences for Latvian: 1) substantially more inflectedness = many more systematically varying word endings; and 2) word roots almost always end with a consonant.

|  |  |
| --- | --- |
| **Language** | **Segmented Text** |
| English | you need to know exactly what you want to im–mortal–ise dur–ing the photo session , and be able to tell the photo–graph–er about it . |
| Latvian | ir jāsaprot , ko tieš–i tu vēl–ies ie–mūž–ināt foto–sesij–as laik–ā un jāpa–stāst–a par to foto–grāf–am . |

PRPE has two phases:

* The **learning phase**, in which ranked lists of main building blocks (potential prefixes, roots and postfixes) are obtained;
* The **application phase**, in which segmentation is performed using obtained building blocks.

From the algorithmic perspective, PRPE contributes two main ideas:

* The ‘Root alignment’ principle to extract potential roots and other sub-words in the learning phase;
* A special technique to construct words from obtained potential sub-words thus accomplishing word segmentation.

## Obtaining Potential Segments

The main goal of the learning phase of PRPE is to obtain lists of potential prefixes, roots and postfixes (suffixes and endings) from a single-language corpus.



**Figure 1**. Illustration of the building blocks used in PRPE for the word “unbelievables”.

The key idea of the algorithm is the ‘Root alignment’ principle (see illustrations in Fig. 1 and Fig. 2, and example of implementation in Fig. 3):

* Left substrings of words are considered potential roots;
* Aligning potential roots with the middle parts of words allows extracting potential prefixes and postfixes.



**Figure 2.** The illustration of the ‘Root alignment’ principle in word “unbelievables”: potential roots aligned with the middle part of the word to collect statistics for prefix “un”.

Obtaining potential segments is carried out in four steps:

1. Collecting frequency statistics of left and right substrings of words. For instance, among the most frequent left substrings in English we can found “the”, “ther”, “re”, “commis”, but among the most popular right substrings – “s”, “es”, “tion”, “ation”.;
2. Extracting potential prefixes from left substrings through aligning other left substrings as potential roots with the middle part of word:
   1. obtain prefix statistics,
   2. select the most frequent prefixes to become potential prefixes in segmentation;
3. Extracting potential postfixes from right substrings through aligning other left substrings as potential roots with the middle part of word:
   1. obtain postfix statistics (in a similar way as for prefixes),
   2. select endings from postfixes according predefined rules to become potential endings in segmentation;
   3. extract and select the most frequent suffixes from postfixes by splitting away collected endings – to become potential suffixes in segmentation;
4. Extracting potential roots from left substrings through aligning them with the middle part of word considering already collected prefixes and postfixes. Here longer roots are also assigned bigger weight coefficients to better compete with smaller roots in the segmentation phase.

All the obtained lists of potential sub-words are ranked, and the predefined hyper-parameters determine how many of the respective sub-words will become final building blocks. Ranking numbers (1, 2, 3, etc.) will be then used to calculate the best segmentation.

As postfixes are split into suffixes and endings (which is not so important for English, but matters for morphologically rich languages), the output of the learning phase consists of four ranked lists: prefixes, roots, suffixes and endings.

|  |
| --- |
| **module** extract\_potential\_prefixes (*vocab*, *leftstat*):  *vocab* – list of all words found in the text corpus  *leftstat* – statistics of frequencies of left substrings  *prefstat* – prefix statistics to be calculated  **for each** word ***w*** in the vocabulary *vocab*:  **for each** left substring ***p*** **in** ***w***: *# a potential prefix*  **if** ***p*** is a valid prefix according to a hardcoded control:  *# a potential root in the middle of w:*  **for each** substring **r** **in** **w** just after **p**:  **if** ***r*** is a valid root according to a hardcoded control  **and** ***r*** is found **in** *leftstat*:  prefstat[**p**] = prefstat[**p**] + leftstat[**r**]  **return** *prefstat* |

**Figure 3**. Prefix extraction module to algorithmically illustrate the ‘Root alignment’ principle: trying to locate potential roots (frequent left substrings) in the middle of a word to extract potential prefixes. Extracting postfixes is designed using the same approach.

## Segmenting Words Using Obtained Potential Segments

The segmentation phase uses ranked lists (prefixes, roots, suffixes and endings) to segment words. Ranking numbers are used to calculate the best segmentation candidate.

Segmenting a word is carried out in the following way:

1. All possible segmentations for the word are obtained;
2. The highest ranked candidate segmentation wins.

### Collecting all possible segmentations. Four ranked lists of potential segments available (P: prefixes, R: roots, S: suffixes and E: endings) for segmentation. Each candidate segmentation is built in the following form:

*([p] [p] r [s] [e]) +,* (1)

where *p*ϵP, rϵR, sϵS, eϵE.

This means that one segmentation is one or more ‘root blocks’ (as root is the only mandatory block in the big block). We search for two prefixes because the two prefix case is quite common in Latvian (an example from English would be “non-re-active”).

Example of segmentation candidates for word “unbelieve” (‘/’ marks boundary of two candidate ‘root blocks’):

* un–bel–ieve
* un–bel–i / eve
* un–believ–e
* un–believe

### Calculating the best segmentation. The best segmentation is the highest ranked segmentation from those with the smallest number of ‘root blocks’, and the rank of the segment is the sum of ranks of individual blocks. In the example above the segmentation #2 is of two ‘root blocks’, i.e., out of competition.

## Named Entity Processing

[TO DO]

## Additional Heuristics

### Several addition heuristics were used to tune the algorithm for better results.

#### Most frequent words unsegmented. To reduce the final number of segments, a predefined number of the most frequent words stay unsegmented (see ‘leave-out rate’ in the results).

#### Optimization of the segmentation. To reduce the final number of segments, several heuristics are used to join back some segments, e.g.:

* prefixes not split away,
* suffixes not split away between roots.

#### No segmentation candidates. If there are no segmentations candidates (i.e., a word cannot be built using available blocks), only the best postfix is split away.

#### Uppercase marking. A word starting with uppercase and the rest symbols in lowercase converted to lowercase and a special uppercase marker is inserted before it.

## Adapting the Algorithm to a Particular Language

As the algorithm is not fully language-independent, some minor adaptation should be carried out for a particular language:

1. Add a small amount of language-specific source code (candidate word parts are additionally screened by a small number of hand-coded routines/rules);
2. Tune hyperparameters (e.g., how many prefixes should be selected as potential prefixes, minimum length of prefixes).

According to the experiments, adapting to a particular language noticeably increases the segmentation quality.

# Experiments and Results

The main idea for the experiments was to show that pre-processing corpora with PRPE yields better machine translation results, relative to baseline segmentation schemes.

For our experiments, we used the English-Latvian dataset provided in the WMT 2017[[5]](#footnote-5) shared task in news translation. The approximate size of each of the parallel corpora – 1.6M sentences. We use as a starting point the data as pre-processed (filtered, normalised, tokenised) by the authors of [12] for their experiments.

We obtained sub-word-segmented versions of both the English and Latvian texts using various configuration of PRPE, as well as two baseline segmentation algorithms:

1. BPE ([3])[[6]](#footnote-6);
2. Tilde’s Morphologically segmented version of the same dataset, also provided to us by the authors of [1], [12];

All the non-BPE segmentations were also post-processed using BPE to better support open-vocabulary translation (by ensuring full coverage of the word vocabulary in the training data, since that is not an explicit goal/guarantee of the alternative segmentation schemes). In all cases, both languages were segmented similarly, using the same algorithm with one set of configuration parameters per experiment.

To evaluate the impact of PRPE on machine translation, we then used these various sub-word-segmented parallel corpora to train English-to-Latvian (en-lv) and Latvian-to-English (lv-en) translation models using two architecturally quite different NMT systems:

1. Nematus ([14])[[7]](#footnote-7) is a framework for NMT based on what has become essentially the standard reference architecture for learning sequence to sequence translation tasks: encoder-decoder using recurrent neural networks, with an attention mechanism that gives the decoder access to richer information about the input sequence than what the encoder can encode into a fixed-length vector. For our primary baseline, we chose a relatively basic, straightforward configuration of the many options supported by Nematus: Hidden layer size=1024, word embedding dimensions=500, batch\_size=60, max length for input sequences=80, no dropout, optimization using Adadelta, with early stopping after loss computed on a cross-validation dataset fails to improve for 10 x 10,000 batches. Default values were used for the depth of the recurrence transitions in the encoder and decoder (= 1 and 2, respectively).
2. ConvS2S (“Convolutional Sequence to Sequence”, [15])[[8]](#footnote-8) is a newer architecture that has recently posted some new state-of-the results for NMT. Instead of recurrent neural networks, it uses convolutional networks for its encoder and decoder, a design choice which allows for greater parallelism when training the model (enabling significantly faster training). For our baseline configuration we simply used one of the default configurations included with the framework: “fconv\_wmt\_en\_ro”, a configuration originally used for an English->Romanian NMT model. It is a fairly deep model, with 20 layers in each of its encoder and decoder neworks, word embedding dimensions=512, hidden layer size=512.

Training even a relatively small NMT model on one or two GPUs takes a minimum of several days, so resource and time constraints precluded our doing much in the way of search over the space of potential configuration and training hyperparameters for the NMT systems we used. But since our goal was not to find optimal configurations and maximize translation BLEU scores, but instead to test for incremental benefits from using our proposed sub-word segmentation scheme, we chose an initial set of NMT configuration and training parameters (yielding reasonably good baseline results), and then used them unchanged for all subsequent experiments. We did, however, try various settings of the internal parameters of the PRPE algorithm, and found that different settings yielded best results for Nematus vs. ConvS2S. This leads to the observation that PRPE configuration should be tuned in concert with other hyperparameters when training an NMT system. (This is completely analogous to selecting the number of merge operations for BPE.) In particular, the “leave-out rate” seems to be the most important tunable parameter for PRPE.

Previous results[[9]](#footnote-9) have shown that the translation direction English-to-Latvian generally yields worse scores than Latvian-to-English, and in all cases our results were consistent with this finding. This could be explained by the supposition that translation towards a morphologically more rich language is a more challenging task. That’s why we hoped to obtain improvements in this particular direction. Unfortunately, with Nematus, the best configuration of PRPE gave a minor (but not statistically significant[[10]](#footnote-10)) improvement in BLEU score for lv-en (Latvian-to-English) translation, but in the en-lv direction produced almost identical scores to the morphologically segmented baseline (see Table 5). With ConvS2S we observed statistically significant improvements in both directions (see Table 6).

**Table 5.** Translation results with Nematus system using various segmentation techniques.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **BPE (BLEU)** | **Tilde’s morph (BLEU)** | **PRPE (leave-out rate = 5000)** | |
| **BLEU** | **p-val vs BPE** |
| en-lv | 17.05 | 17.15 | **17.16** | 0.23 |
| lv-en | 18.66 | 18.67 | **18.90** | 0.13 |

**Table 6.** Translation results with ConvS2S system using various segmentation techniques.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | BPE (BLEU) | Tilde’s morph (BLEU) | PRPE (leave-out rate = 5000) | |
| BLEU | p-val vs BPE |
| en-lv | 20.30 | 21.26 | **21.33** | 0.00 |
| lv-en | 21.93 | 22.05 | **22.61** | 0.01 |

**Table 7.** Translation results using deeper Nematus models.

|  |  |  |  |
| --- | --- | --- | --- |
|  | BPE (BLEU) | PRPE (leave-out rate = 5000) | |
| BLEU | p-val vs BPE |
| en-lv | 19.13 | **19.55** | 0.06 |
| lv-en | 20.90 | **21.46** | 0.01 |

Note that the baseline scores that we obtained using ConvS2S were 3-4 BLEU points higher than the corresponding scores obtained using Nematus with the same datasets. We conjecture that this might be to a large extent because we were using a relatively basic (shallow) configuration of Nematus, with less modeling capacity than the large and deep default configuration we chose for ConvS2S. To test this conjecture – and the possibility that deeper networks might be better able to make use of more sophisticated sub-word segmentation schemes (as suggested by the bigger boost from PRPE that we saw with ConvS2S vs. Nematus) – we ran a few additional experiments using a configuration for Nematus based on training scripts provided by Edinburgh University[[11]](#footnote-11) [16], which make use of some Nematus features that allow for using deeper network configurations in its encoder and decoder [17]. Initial results (see Table 7) seem to confirm these conjectures, but, due to time constraints, a more systematic exploration will have to await future work.

# Conclusion

In this paper, we propose an algorithm for close-to-morphological word segmentation for machine translation without requiring the availability of language specific morphologically labelled data. Experimental results demonstrated that PRPE pre-processing of training data for NMT can yield small improvements in translation output, relative to pre-processing with baseline sub-word segmentation algorithms. But the results also show that machine translation with inflected languages remains a big challenge, especially with translation direction towards a highly inflected language.

The PRPE algorithm exploits the ‘Root alignment’ principle to extract potential sub-words, as well as a special technique to construct words from potential sub-words.

In addition, the experiments showed that fully splitting all affixes is counter-productive, in that it produces too long sequences of sub-words, and the translation quality grows worse. The best results were achieved with only compound splitting plus splitting postfixes from the end of a word, as well as leaving up to 5000 of the most frequently encountered words unsegmented.

Obtained improvements in translation quality from PRPE pre-processing were not particularly large, in some cases falling below a commonly used threshold for statistical significance, which might be a signal that the approach of autonomous (without using syntactic and semantic context) pre-processing to do sub-word segmentation might be near its limits for potential improvements.

Other already started experiments beyond the scope of this paper include running experiments training NMT models for other language pairs, as well as using parallel corpora to improve PRPE segmentation.

# Acknowledgements

The research has been supported by the European Regional Development Fund within the research project “Neural Network Modelling for Inflected Natural Languages” No. 1.1.1.1/16/A/215, and the Faculty of Computing, University of Latvia.

# References

[1] M. Pinnis, R. Krišlauks, D. Deksne, T. Miks, Neural Machine Translation for Morphologically Rich Languages with Improved Sub-word Units and Synthetic Data. *Ekštein K., Matoušek V. (eds) Text, Speech, and Dialogue. TSD 2017. Lecture Notes in Computer Science*, *vol. 10415*. Springer, Cham (2017).

[2] T. Ruokolainen, O. Kohonen, K. Sirts, A. Grönroos, M. Kurimo, S. Virpioja, A Comparative Study of Minimally Supervised Morphological Segmentation. *Computational Linguistics* **42**, issue 1(2016), 91-120.

[3] R. Sennrich, B. Haddow, A. Birch, Neural Machine Translation of Rare Words with Subword Units. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL2016)* **1**, (2016), Berlin, Germany, 1715–1725.

[4] K. Papineni, S. Roukos, T. Ward, T., Zhu, W.J. BLEU: a method for automatic evaluation of machine translation. *ACL '02 Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, (2002), Philadelphia, Pennsylvania, 311–318.

[5] J. Hajič, Morphological Tagging: Data vs. Dictionaries. *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference (NAACL 2000)* (2000), Seattle, Washington, 94-101.

[6] P. Paikens, L. Rituma, L. Pretkalnina, Morphological analysis with limited resources: Latvian example. *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA)* (2013), Linköping, Sweden, 267-277.

[7] M. Pinnis, K. Goba, *Maximum Entropy Model for Disambiguation of Rich Morphological Tags*. International Workshop on Systems and Frameworks for Computational Morphology (2001), Springer, Berlin, Heidelberg, 14-22.

======= 8 through 11 deleted ==========================================

[12] M. Pinnis, R. Krišlauks, T. Miks, D. Deksne, V. Šics, Tilde’s Machine Translation Systems for WMT 2017. *Proceedings of the Second Conference on Machine Translation (WMT 2017)* (2017), Copenhagen, Denmark, 374–381.

======= 13 deleted ==========================================

[14] R. Sennrich, O. Firat, K. Cho, A. Birch, B. Haddow, J. Hitschler, M. Junczys-Dowmunt, S. Läubli, A.V.M. Barone, J. Mokry, M. Nadejde, Nematus: a Toolkit for Neural Machine Translation. *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics* (2017), Valencia, Spain, 65-68.

[15] J. Gehring, M. Auli, D. Grangier, D. Yarats, Y. Dauphin, Convolutional Sequence to Sequence Learning. *Proceedings of the 34th International Conference on Machine Learning* (2017), Sydney, Australia, 1243-1252.

[16] R. Sennrich, A. Birch, A. Currey, U. Germann, B. Haddow, K. Heafield, A.V.M. Barone, P. Williams, The University of Edinburgh’s Neural MT Systems for WMT17. *Proceedings of the Second Conference on Machine Translation,* **2**: Shared Task Papers (2017). Copenhagen, Denmark, 389-399.

[17] A.V.M. Barone, , J. Helcl, R. Sennrich, B. Haddow, A. Birch, Deep Architectures for Neural Machine Translation. *Proceedings of the Second Conference on Machine Translation* (2017), Copenhagen, Denmark, 99-107.

[trans2017] Zhongwei Li ; Eng Siong Chng ; Haizhou Li. Named entity transliteration with sequence-to-sequence neural network

1. Jānis Zuters, [0000-0002-3194-9142], e-mail: janis.zuters@lu.lv [↑](#footnote-ref-1)
2. Gus Strazds, [0000-0002-2761-8586], e-mail: gstrazds@gmail.com [↑](#footnote-ref-2)
3. Source code available at: https://github.com/zuters/prpe [↑](#footnote-ref-3)
4. Source code available at: https://github.com/zuters/prpene [↑](#footnote-ref-4)
5. http://www.statmt.org/wmt17/translation-task.html [↑](#footnote-ref-5)
6. https://github.com/rsennrich/subword-nmt [↑](#footnote-ref-6)
7. https://github.com/EdinburghNLP/nematus [↑](#footnote-ref-7)
8. https://github.com/facebookresearch/fairseq-py [↑](#footnote-ref-8)
9. http://www.statmt.org/wmt17/results.html [↑](#footnote-ref-9)
10. Statistical significance was estimated via bootstrap resampling using the script analysis/bootstrap-hypothesis-difference-significance.pl from the Moses MT system: https://github.com/moses-smt/mosesdecoder [↑](#footnote-ref-10)
11. http://data.statmt.org/wmt17\_systems/training [↑](#footnote-ref-11)