PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 109 OCTOBER 2017 39–50

Visualizing Neural Machine Translation Attention and Confidence

Matīss Rikters,[a] Mark Fishel,[b] Ondřej Bojar[c]

[a] Faculty of Computing, University of Latvia
[b] Institute of Computer Science, University of Tartu
[c] Charles University, Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics

Abstract

In this article, we describe a tool for visualizing the output and attention weights of neural machine translation systems and for estimating confidence about the output based on the attention.

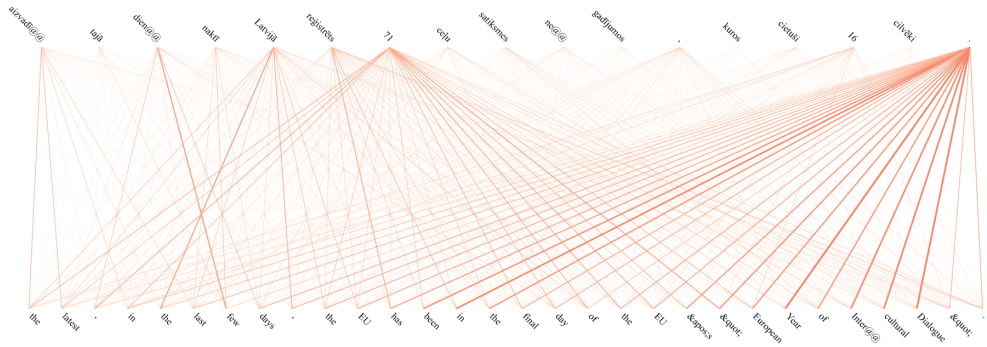
Our aim is to help researchers and developers better understand the behaviour of their NMT systems without the need for any reference translations. Our tool includes command line and web-based interfaces that allow to systematically evaluate translation outputs from various engines and experiments. We also present a web demo of our tool with examples of good and bad translations: `http://ej.uz/nmt-attention`.

1. Introduction

The world of machine translation (MT) is in transition between the well-recognized statistical MT (SMT, Koehn, 2009) and the new and exciting neural MT (NMT, e.g. Bahdanau et al., 2014). While the systems themselves are slowly being replaced, the necessities behind analyzing them remain the same, as do the tools built mostly for the older approaches.

In this paper, we introduce a translation inspection tool that specifically targets NMT output. The tool uses the attention weights corresponding to specific token pairs during the decoding process, by turning them into one of several visual representations that can help humans better understand how the output translations were produced. The tool also uses the attention information to estimate the confidence in

 Corresponding author: matiss@lielakeda.lv
Cite as: Matīss Rikters, Mark Fishel, Ondřej Bojar. Visualizing Neural Machine Translation Attention and Confidence. The Prague Bulletin of Mathematical Linguistics No. 109, 2017, pp. 39–50.
doi: 10.1515/pralin-2017-0037.



Source:	Aizvadītajā diennaktī Latvijā reģistrēts 71 ceļu satiksmes negadījumos, kuros cietuši 16 cilvēki.
Hypothesis:	The latest, in the last few days, the EU has been in the final day of the EU's "European Year of Intercultural Dialogue".
Reference:	71 traffic accidents in which 16 persons were injured have happened in Latvia during the last 24 hours.

*Figure 1. A Latvian to English neural translation output that has no relation to the input. The weak connection is obvious from the visualized attention weights, even without knowing the source and target languages or seeing the input or output texts. Confidence: **18.11%**; CDP: 44.49%; APout: 67.41%; APin: 79.58%.*

translation which allows to distinguish acceptable outputs from completely unreliable ones, no reference translations are required.

The paper is structured as follows: Section 2 summarizes related work on tools for inspecting translation outputs and alignments. Section 3 describes the tool from the users' point of view, covering the web-based and command-line visualizations and the confidence score for better navigation. Section 4 provides a look into the back-end of the system. Finally, conclusions and future work directions are in Section 5.

2. Related Work

Zeman et al. (2011) describe Addicter—a set of command-line and simple web-based tools that can be useful for inspecting automatic translations and finding systematic errors among them. One of the tools in Addicter, *alitextview.pl*, is designed to convert SMT alignments from the typical alignment pair format (*source_token_id – target_token_id*) to a table representation, making it more human-readable. Our command-line interface took much inspiration from this work while adapting to the specifics of the NMT counterpart of alignments.

Madnani (2011) introduces iBLEU—a web-based tool for visualizing BLEU (Papineni et al., 2002) scores. Unlike alignments between the source and the hypothesis, the calculation of BLEU requires a reference translation to which the hypothesis will be compared. On top of that, iBLEU also allows to add another file with hypotheses from another MT system for a direct comparison. Given these inputs, the tool highlights the differences between the translations and reference material. It also enables easy navigation through the set of sentences by representing the BLEU score of each sentence in a clickable bar chart. A quick jump to a specific sentence is possible by entering its number. The clickable chart and jumps seemed most desirable features for us, so we added similar capabilities to the web version of our tool.

Klejch et al. (2015) developed MT-ComparEval—a web-based translation visualization tool that seems to build upon iBLEU by adding many more fine-grained features. It also allows to compare differences between translations and references, other translations and the source input. The main differences are that (1) MT-ComparEval stores all imported data as experiments for viewing at any time, where iBLEU forgets everything upon a page refresh; (2) for each of these experiments, one can add output from multiple systems (iBLEU can cope with only 2); (3) MT-ComparEval displays additional scores (precision, recall, F-measure); and (4) it shows various detailed sentence and n-gram level statistics with configurable highlighting of the differences. A noticeable shortcoming is that one cannot jump to a specific sentence in the set. While ordering by sentence ID is possible, to view the 1000th of 2000 one would have to scroll through the first 999.

Nematus (Sennrich et al., 2017) includes a set of utilities for visualizing NMT attentions. The first one, *plot_heatmap.py* plots alignment matrices similar to the previously mentioned *alitextview.pl*, using Nematus output translations with alignments. The second tool, *visualize_probs.py* generates HTML for a web view that displays the output translation in a table with the background of each token shaded according to the attention weight. The final tool, consisting of *attention.js* and *attention_web.php*, connects source and target tokens with lines as thick as the corresponding attention weights between them. However there is no tool included to generate the latter visualization for an arbitrary sentence - it is given only in the form of one set example. This last tool was a strong inspiration for building our tool. We reused parts of its code in the web version of our visualization.

Neural Monkey (Helcl and Libovický, 2017) provides several visualization tools for checking the training process that include visualizing attention as soft alignments. It can generate matrices similar to the previously mentioned *alitextview.pl* for each sentence in the first validation batch during the training process. A few drawbacks of this method are that the images are (1) of a static size (the predefined maximum input length * maximum output length) - if sentences are longer, the attention image gets cut off, if shorter, bottom rows of the matrix (representing the input) are left black and columns (representing the output) on the far right side are filled with "phantom" attention; (2) no input and output words, tokens or subword units are displayed, only

the matrix; (3) there is no option to generate visualizations for a test set outside the system training process.

3. The Tool from Users' Perspective

The main goals of our tool are to provide multiple ways of visualizing NMT attention alignments, as well as to make it easy to navigate larger data sets and find specific examples. To accomplish these goals, we implemented two main variations of our tool, a textual command line visualization and a web-based visualization. This chapter provides an insight into the features of both of them and suggestions as to when they can be useful.

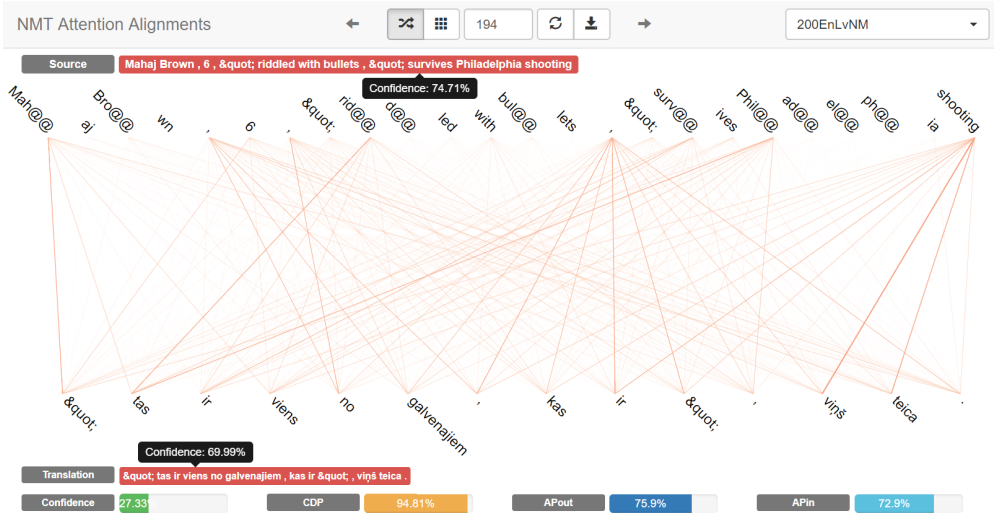
3.1. Web Browser Visualization

The web visualization is intended to provide an intuitive overview of one or multiple translated test sets. This is done by showing one sentence at a time, with navigation to other sentences by ID, length or multiple confidence measures. Switching between experiments (test sets) is also easy. For each individual sentence, four confidence metrics are shown, and a confidence score for each source and translated token (or subword unit). The tool also allows to export the alignment visualization of any selected sentence to a high-resolution PNG file with one click.

The essential part of the visualization is presented in the following way: source tokens (at the top) are connected to translated tokens (at the bottom) via orange lines, ranging from completely faint to very thick, as shown in Figures 2 and 3. A thicker line from a translated token to a source token means that the decoder paid more attention to that source token when generating the translation. Ideally, these lines should mostly be thick with some thinner ones in between. When they look chaotic, connecting everything to everything (Figure 2) or everything in the translation is connected to mostly a single token in the source,[1] that can be well an indication of an unsuccessful translation that will possibly have little or no relation with the source sentence. On the other hand, if all lines are thick, straight downwards, connected one-to-one (see the right part of Figure 3), that may point to nothing being translated at all.

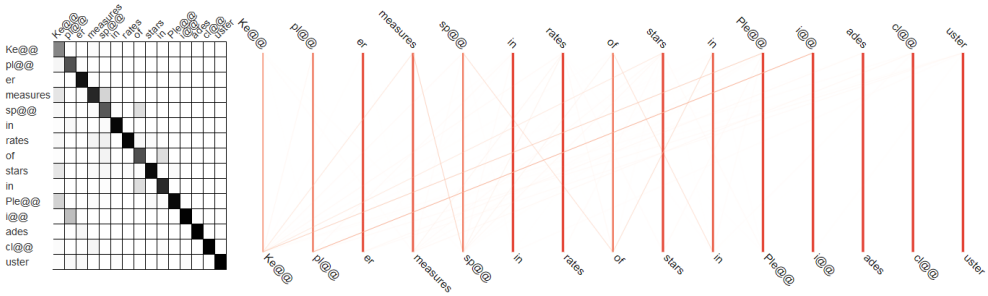
Additionally, the matrix style visualization is also available in the web version as shown on the left part of Figure 3.

[1] Such tokens were called "garbage tokens" in IBM-style word-alignment methods (Och and Ney, 2000), and they were often rare words where the model had the option to attribute everything to them.



Source: Mahaj Brown , 6 , ”riddled with bullets ,” survives Philadelphia shooting
Hypothesis: ”tas ir viens no galvenajiem , kas ir” , viņš teica.
Reference: 6 gadus vecais Mahajs Brauns ”ložu sacaurumots” izdzīvo apšaudē Filadelfijā.

*Figure 2. An example of a translated sentence that exhibits a low confidence score. Confidence: **27.33%**; CDP: 94.81%; APout: 75.9%; APin: 72.9%.*



Source: Kepler measures spin rates of stars in Pleiades cluster
Hypothesis: Kepler measures spin rates of stars in Pleiades cluster
Reference: Keplers izmēra zvaigžņu griešanās ātrumu Plejādes zvaigznājā.

Figure 3. An example of a translated sentence that exhibits a suspiciously high confidence score. The translation here is a verbatim rendition of the input. Matrix form visualization on the left, line form visualization on the right. Confidence: ***95.44%****; CDP: 100.0%; APout: 98.84%; APin: 98.85%.*

3.2. Confidence Scores

To aid in locating suspicious and potentially bad translations, we introduced a set of confidence metrics (more details in Section 4.1). For each sentence, the tool displays an overall confidence score, coverage deviation penalty, and input and output absentmindedness penalties. The overall confidence score is also shown for each source token, indicating the amount of confidence that the token has been used to generate a correct translation, as well as for each translated token, indicating the amount of confidence that it is a correct translation. All of these scores are represented in percentages from 0 to 100 and can be used to navigate through the test set (Figure 4), making it easy to quickly find very good or very bad translations among hundreds. The selected sentence is highlighted simultaneously across all navigation charts and each chart can be sorted in either direction or reset to the order by sentence ID.

3.3. Command Line Visualization

The command line visualization is available in three different formats: (1) using twenty-five different shades of gray as shown in Figure 5; (2) using five gradually shaded Unicode block elements as shown in Figure 6; and (3) using nine gradually filled Unicode block elements. Each sentence is output via a graphical matrix, where rows represent the source input tokens or subword units and columns representing the target side. The corresponding tokens are printed out on the bottom (target) or far right side (source) of the matrix. Unlike the authors of *alitextview.pl*, we chose

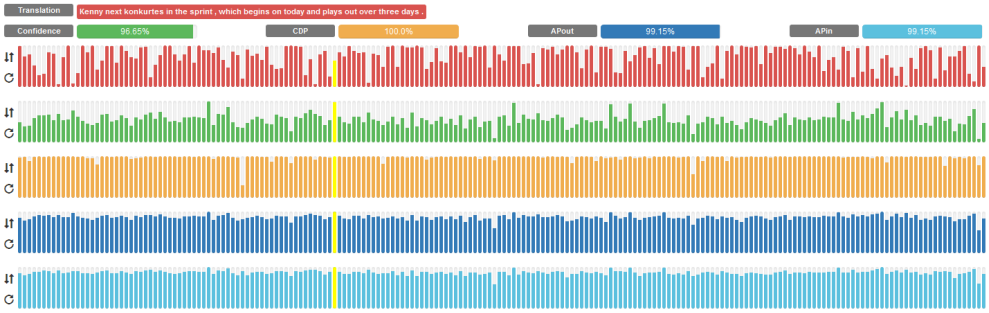


Figure 4. Navigation charts allow to jump to a sentence based on its length in characters (red), confidence (green), coverage deviation penalty (dark yellow), absentmindedness penalty for input (dark blue) and output (light blue). The currently active sentence is highlighted in bright yellow. All charts are sortable and scrollable for a better user experience.

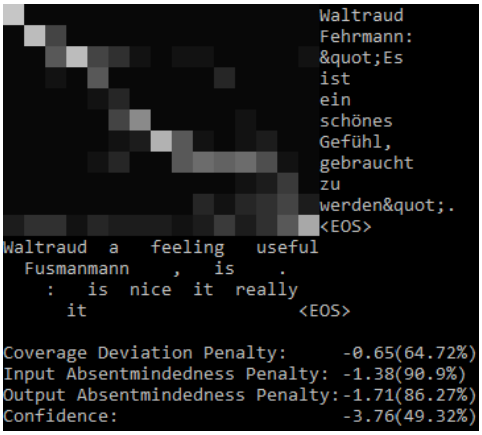


Figure 5. Visualization in the command line, using twenty-five different tones of gray.

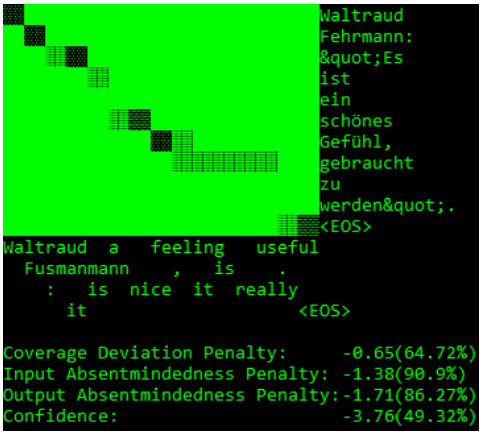


Figure 6. Visualization in the command line, using five differently shaded block elements.

to represent the source tokens on the right, so that the graphical matrix starts at the beginning of the line for each sentence. After each sentence, one empty line is printed.

One obvious use case for the command-line visualization is to directly compare alignments of NMT attention with the ones produced by SMT. This type of visualization is also the fastest, therefore it can be used to quickly check alignments for a specific sentence. Fixed-width Unicode fonts can be used in almost all text editors, so redirecting output in the *block* mode to a text file to share with others is also a useful application. However, to view the *color* version from a text file, it needs to be interpreted as xterm color sequences, e.g. using "`less -R`" in a Linux terminal.

4. System Description

The visualization tool is developed in Python and PHP. It is published in a GitHub repository[2] and open-sourced with the MIT License.

Both visualizations can be run directly from the command line. The web version is capable of launching on a local machine without the requirement for a dedicated web server.

4.1. Scoring Attention

This section provides details about how the previously mentioned confidence scores are calculated and outlines what is needed to make good use each option.

The basis of our scoring methods was influenced by Wu et al. (2016), who defined a coverage penalty for punishing translations that do not pay enough attention to input tokens:

$$\mathrm{CP} = \beta \sum_{j} \log\left(\min\left(\sum_{i} \alpha_{ji}, 1.0\right)\right), \tag{1}$$

where CP is the coverage penalty, i is the output token index, j is the input token index and β is used to control the influence of the metric. To complement that, we introduce a set of our own metrics:

- **Coverage Deviation Penalty** (CDP) penalizes attention deficiency and excessive attention per input token.
- **Absentmindedness Penalties** ($\mathrm{AP}_{\mathrm{out},\,\mathrm{in}}$) penalize output tokens that pay attention to too many input tokens, or input tokens that produce too many output tokens.
- **Confidence** is the sum of the three metrics – CDP, $\mathrm{AP}_{\mathrm{in}}$ and $\mathrm{AP}_{\mathrm{out}}$.

Coverage Deviation Penalty

Unlike CP, CDP penalizes not just attention deficiency but also excessive attention per input token. The aim is to penalize the sum of attentions per input token for going

[2]NMT Attention Alignment Visualizations: `https://github.com/M4t1ss/SoftAlignments`

too far from 1.0, so that tokens with the total attention of 1.0 get a score of 0.0 on the logarithmic scale, while tokens with less attention (like 0.13) or more attention (like 3.7) get lower values. We thus define the coverage deviation penalty:

$$\mathrm{CDP} = -\frac{1}{J}\sum_{j}\log\left(1 + \left(1 - \sum_{i}\alpha_{ji}\right)^{2}\right). \tag{2}$$

The metric is on a logarithmic scale, and it is normalized by the length J of the input sentence in order to avoid assigning higher scores to shorter sentences.

Absentmindedness Penalties

To target scattered attention per output token, we introduce an absentmindedness penalty:

$$\mathrm{AP}_{out} = -\frac{1}{I}\sum_{i}\sum_{j}\alpha_{ji}\cdot\log\alpha_{ji}. \tag{3}$$

It evaluates the dispersion via the entropy of the predicted attention distribution, resulting in values from 1.0 for the lowest entropy to 0.0 for the highest. The values are again on the log-scale and normalized by the source sentence length I.

The absentmindedness penalty can also be applied to the input tokens after normalizing the distribution of attention per input token:

$$\mathrm{AP}_{in} = -\frac{1}{I}\sum_{j}\sum_{i}\alpha_{ij}\cdot\log\alpha_{ij}. \tag{4}$$

The final confidence score sums up all three above mentioned metrics:

$$confidence = \mathrm{CDP} + \mathrm{AP}_{out} + \mathrm{AP}_{in}. \tag{5}$$

For visualization purposes each of the scores needed to be set on the same scale of 0-100%. To achieve that, we applied

$$\mathrm{percentage} = e^{-C(X^2)}, \tag{6}$$

where X is the score to convert and C is a constant of either 1 for CDP or 0.05 for the other scores (AP_{out}, AP_{in}, confidence). Other constants were also tested, but these specific ones seemed to best fit data from our test sets, by displaying the percentage values across the whole range.

4.2. System Architecture

The code can be divided into two logical parts - 1) processing input data and generating output data and 2) displaying and navigating the generated output data in a web

browser. The former part is written in Python and handles all input data, generates output data, displays the command line visualization or launches a temporary web server for the web browser visualization. Each time a web visualization is launched, a new folder is created within */web/data* where all necessary output data files are stored, a temporary PHP web server is launched on *127.0.0.1:47155*, and the address is opened as a new tab in the default web browser. After stopping the script all data remains in the */web/data* and can be accessed later as well.

The latter part is responsible for everything that is shown in the browser. It mainly consists of PHP, HTML and JavaScript code that facilitates quick navigation between sentences even in larger data files, as well as navigation charts and sorting, visualization export to image files and a responsive user interface. If necessary, this part can be used as a stand-alone website for displaying and interacting with pre-generated results.

4.3. Requirements and Usage

The requirements are as follows:

- Python (2 or 3) and NumPy,
- PHP 5.4 or newer (for web visualization),
- Nematus or Neural Monkey (for training NMT systems),
- Nematus, AmuNMT[3] (Junczys-Dowmunt et al., 2016) or Neural Monkey (for translating and extracting attention data)
  - Or any NMT framework that can output an attention matrix for each translation (may require format conversion) .

To use the tool, first translate a set of sentences using a supported NMT framework with the option of saving alignments[4] switched on. The sources combined with the resulting translations and attention matrices can then be used as input for the *process_alignments.py* script. Depending on the selected output type, alignments will either be displayed in the terminal or a new tab will be opened in the default web browser. Example input files from each supported NMT framework are provided along with commands to run them.

5. Conclusions

In this paper, we described our tool for visualizing attention alignments generated by neural machine translation systems and for estimating confidence of the translation. The tool aims to help researchers better understand how their systems perform by enabling to quickly locate better and worse translations in a bigger test set. Compared to other similar tools, ours relies on the confidence scores and does not require

[3]Barvins/amunmt (forked from marian-nmt/marian): `https://github.com/barvins/amunmt`

[4]How to get alignment files from NMT systems: `https://github.com/M4t1ss/SoftAlignments`

reference translations to facilitate this easier navigation. This allows to integrate it, for example, in an NMT system with a web interface, providing users with an explanation for the result of a specific translation.

In the future, we plan to integrate a part of this tool into one public NMT system, Neurotolge.[5] We will also extend the out-of-the-box support to other popular NMT frameworks like OpenNMT[6] or tensor2tensor.[7]

Acknowledgements

A part of this research was supported by the ICT COST Action IC1207 *ParseME: Parsing and multi-word expressions. Towards linguistic precision and computational efficiency in natural language processing*, the grant H2020-ICT-2014-1-645442 (QT21) and Charles University Research Programme "Progres" Q18+Q48.

The authors would like to thank Mārcis Pinnis and Raivis Skadiņš for advice, comments and suggestions. Also, Pēteris Ņikiforovs for the base code of the web-based matrix visualization.

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, abs/1409.0473, 2014. URL `http://arxiv.org/abs/1409.0473`.

Helcl, Jindřich and Jindřich Libovický. Neural Monkey: An Open-source Tool for Sequence Learning. *The Prague Bulletin of Mathematical Linguistics*, (107):5–17, 2017. ISSN 0032-6585. doi: 10.1515/pralin-2017-0001. URL `http://ufal.mff.cuni.cz/pbml/107/art-helcl-libovicky.pdf`.

Junczys-Dowmunt, Marcin, Tomasz Dwojak, and Hieu Hoang. Is Neural Machine Translation Ready for Deployment? A Case Study on 30 Translation Directions. In *Proceedings of the 9th International Workshop on Spoken Language Translation (IWSLT)*, Seattle, WA, 2016. URL `http://workshop2016.iwslt.org/downloads/IWSLT_2016_paper_4.pdf`.

Klejch, Ondřej, Eleftherios Avramidis, Aljoscha Burchardt, and Martin Popel. MT-ComparEval: Graphical evaluation interface for Machine Translation development. *The Prague Bulletin of Mathematical Linguistics*, 104(1):63–74, 2015.

Koehn, Philipp. *Statistical Machine Translation*. Cambridge University Press, 2009.

Madnani, Nitin. iBLEU: Interactively debugging and scoring statistical machine translation systems. In *Semantic Computing (ICSC), 2011 Fifth IEEE International Conference on*, pages 213–214. IEEE, 2011.

[5]Tartu University Translator: `http://neurotolge.ee`

[6]OpenNMT: Open-Source Neural Machine Translation: `https://github.com/OpenNMT/OpenNMT`

[7]T2T: Tensor2Tensor Transformers: `https://github.com/tensorflow/tensor2tensor`

Och, Franz Josef and Hermann Ney. A Comparison of Alignment Models for Statistical Machine Translation. In *Proceedings of the 17th conference on Computational linguistics*, pages 1086–1090. Association for Computational Linguistics, 2000. ISBN 1-555-55555-1.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

Sennrich, Rico, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, et al. Nematus: a Toolkit for Neural Machine Translation. *EACL 2017*, page 65, 2017.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, abs/1609.08144, 2016. URL `http://arxiv.org/abs/1609.08144`.

Zeman, Daniel, Mark Fishel, Jan Berka, and Ondřej Bojar. Addicter: What Is Wrong with My Translations? *The Prague Bulletin of Mathematical Linguistics*, 96:79–88, 2011.

Address for correspondence:
Matīss Rikters
`matiss@lielakeda.lv`
Rainis blvd. 19, Riga, Latvia