PosEdiOn: Post-Editing Assessment in PythOn

Antoni Oliver
Universitat Oberta de Catalunya
aoliverg@uoc.edu

Sergi Alvarez
Universitat Pompeu Fabra
salvarezvid@uoc.edu

Toni Badia
Universitat Pompeu Fabra
toni.badia@upf.edu

Abstract

There is currently an extended use of post-editing of machine translation (PEMT) in the translation industry. This is due to the increase in the demand of translation and to the significant improvements in quality achieved in recent years. PEMT has been included as part of the translation workflow because it increases translators' productivity and it also reduces costs. Although effective post-editing requires sufficiently high quality MT output, usual automatic metrics do not always correlate with post-editing effort. We describe a standalone tool designed both for industry and research that has two main purposes: to collect sentence-level information from the post-editing process (e.g. post-editing time and keystrokes) and to visually present multiple evaluation scores so they can be easily interpreted by a user.

1 Introduction

Post-editing of machine translation (PEMT) is a very common practice in the translation industry. It has been included as part of the translation workflow because it increases productivity when compared with human translation (Aranberri et al., 2014) and reduces costs (Guerberof, 2009) without having a negative impact on quality (Plitt and Masselot, 2010). Post-editors "edit, modify and/or correct pre-translated text that has been processed by an MT system from a source language into (a) target language(s)" (Allen, 2003, p. 296).

In the last few years, both research and industry have become very interested in neural machine translation (NMT) because it has produced very successful results in terms of quality, for example in WMT 2017 (Bojar et al., 2017), WMT 2018 (Bojar et al., 2018) and WMT 2019 (Barrault et al., 2019). Given the overall performance of NMT, it is necessary to study all the potential this approach can offer to post-editing. One of the main problems is that automatic scores give a general idea of the MT output quality but do not always correlate with post-editing effort (Koponen, 2016; Shterionov et al., 2018). Many professional translators state that if the quality of the MT output is not good enough, they delete the remaining segments and translate everything from scratch (Parra Escartín and Arcedillo, 2015).

One of the main goals both of industry and research is to establish a correlation between the quality measurements of the MT output and translators' performance. Research is especially focused on the effort this activity entails, mainly taking into account the temporal, technical, and cognitive effort (Krings, 2001). The use of tools that can log these three dimensions becomes a paramount challenge for research.

Professional translators usually use commercial products to translate and post-edit. In the 2018 Language Industry Survey[1] conducted by EUATC, Elia, FIT Europe, GALA and LINDWeb, SDL Trados[2] was the most used product with more than half of the market quota, followed by MemoQ,[3]

[1] http://fit-europe-rc.org/wp-content/uploads/2019/05/2018-Language-Industry-Survey-Report.pdf
[2] https://www.sdl.com/
[3] https://www.memoq.com

Martins, Moniz, Fumega, Martins, Batista, Coheur, Parra, Trancoso, Turchi, Bisazza, Moorkens, Guerberof, Nurminen, Marg, Forcada (eds.)
Proceedings of the 22nd Annual Conference of the European Association for Machine Translation, p. 403–410
Lisboa, Portugal, November 2020.

Memsource,[4] Wordfast,[5] and Across.[6] However, these existing post-editing environments have a restricted availability and flexibility. As proprietary tools, they are difficult to modify and do not usually provide translator activity data that may be used to study post-editing effort. However, other open-source computer-assisted translation (CAT) environments such as OmegaT,[7] have been modified and used for data collection (Moran et al., 2014).

Instead of trying to reproduce the working conditions of translators, which vary greatly among individuals, other tools establish controlled conditions in order to obtain non-biased data. For this purpose, translators use a post-editing tool that records the post-editing information, can be easily accessed from any platform and has an easy-to-use interface.

In this paper we present PosEdiOn, a simple standalone tool that allows post-editing of MT output and records information of the post-editing effort (time and keystrokes) at sentence-level. It also includes multiple evaluation scores that the user can interpret easily to assess the post-editing process (such as edit distance, HBLEU and HTER). As it does not depend on any specific CAT tool, it allows the collection of post-editing data in a controlled way. It can be used by professionals to assess the convenience of post-editing a certain MT output and by researchers to study post-editing effort.

In Section 2 we analyze some of the previous tools developed for this purpose. The tool and its mains characteristics are presented in Section 3. In Section 4 we describe the PosEdiOn analyzer, which is used to perform all the analysis, and Section 5 includes the conclusions and future work.

2 Previous Work

In order to analyze the different components of post-editing effort, it becomes paramount to use tools that are able to log time, keyboarding, and other potential indicators of cognitive effort (e.g. gaze data). Currently there is a proliferation of these tools (Vieira, 2013), mainly because each research project has specific requirements.

Some of the tools developed focus more on productivity as part of an industry scenario. For example, the Qualitivity[8] plugin can be added to SDL Trados to measure post-editing effort. Alternatively, TAUS developed DQF,[9] which can be used as a standalone benchmark or as an SDL Trados plugin. There has also been EU-funded research to develop open-source workbenches to help improve quantitative measurements of effort (CASMACAT[10] and Matecat[11]).

Other tools collect gaze data, which can be used to study post-editing effort. Tobii Pro Lab is the commercial Windows-oriented eye-tracking software that accompanies Tobii eye trackers. It can calculate a variety of eye-tracking metrics and create visual representations of the data.

Another similar product is Translog-II (Carl, 2012), which is a Windows-oriented program that records user activity data (UAD), that is, all the keystrokes and gaze movements. It is meant specifically for translation process research (TPR) and it offers the possibility of further processing the data with the scripts included in the TPR database of the Centre for Research and Innovation in Translation and Technology (CRITT TPR-DB). Even though these tools collect extensive information, they have specific and demanding settings which are not suitable for all experiments.

Some products devised for a specific experiment are not made available to the public afterwards (Plitt and Masselot, 2010; Green et al., 2013). Other tools focus on obtaining as much information as possible with an easy-to-use product. For example, TransCenter (Denkowski and Lavie, 2012) is an open-source, web-based tool that allows users to carry out PE tasks and logs time and keyboard/mouse activity at a sentence level.

Another tool useful for quantitative investigations specifically designed for post-editing is PET (Aziz et al., 2012). It can also be accessed from any platform, although it is based in Java, which can sometimes be challenging for end-users who need to open the tool from their desktop computers. In addition to recording time and effort indicators at a segment level, PET also allows users to perform evaluation tasks on different customizable scales and criteria. The data file with all the information is saved in xml. However, it does not offer graphics or any other visual information with

[4]https://www.memsource.com
[5]https://www.wordfast.net
[6]https://www.across.net
[7]https://omegat.org
[8]https://appstore.sdl.com/language/app/qualitivity/612/
[9]https://www.taus.net/dqf
[10]https://www.casmacat.eu
[11]https://www.matecat.com

the results nor does it include an analyzer which produces multiple automatic metrics.

3 PosEdiOn

PosEdiOn is a post-editing tool developed mainly to collect information on different implicit and explicit effort indicators. It records time and keystrokes, and it also calculates some of the main indirect effort estimation measures (HTER, HBLEU and edit distance). It produces a file with the raw measurements but it also includes a results file with visually structured information that can be easily understood by any user.

It was developed completely in Python3 and it works in any platform which has Python installed. Translators tend to work from home with a great variety of platforms and devices, and do not always have the computer skills to solve any compatibility errors they may encounter with the tools they are about to use. A Windows executable file is also available, which allows to run PosEdiOn without the need of installing the Python interpreter.

3.1 Files and tasks

PosEdiOn is designed to facilitate the distribution of post-editing tasks in an easy and error-free way. The user receives a zip compressed folder with all the needed elements:

- The PosEdiOn program itself, usually as a Python file. Optionally, a Windows executable can be also used. In this case, sending the zipped file by e-mail can cause problems as some mail providers block attachments with executable files. Alternatively, a link to the zipped file can be used to distribute the post-editing tasks.

- The configuration file (*config.yaml*) that provides all the information necessary for the post-editing task. See section 3.3.

- The post-editing task itself as a tab delimited plain text file. The text file is structured in four fields: source text, machine translated text, post-edited text and segment status.

For translation tasks, only the first field is compulsory. In this case, the translator will be presented only with the source text. For post-editing tasks, the first two fields are compulsory and the post-editor will be presented with the source text and the output text from MT. Each time a segment is validated, this file and the status of the segment are updated.

Once the compressed file is received, it must be unzipped. After executing the program, the task is directly presented. When the translator begins to work on the new task, a new file (*actions.txt* or any other file name stated in the configuration file) is created. All actions including keystrokes, mouse actions and button clicks are stored in this file along with the time it is performed. An example can be seen in the following figure:

```
START 1 2020-02-22 22:28:04.979308
F 1 2020-02-22 22:28:04.996692 Focus_in
M 1 2020-02-22 22:28:08.840216 Mouse.button1
F 1 2020-02-22 22:28:08.840857 Focus_in
K 1 2020-02-22 22:28:09.742533 Key.letter.u 1.6
M 1 2020-02-22 22:28:13.129137 Mouse.button1
OUT 1 2020-02-22 22:28:23.827548
IN 2 2020-02-22 22:28:23.829034
K 2 2020-02-22 22:28:25.018297 Command.CtrlReturn 1.8
OUT 2 2020-02-22 22:28:25.020480
IN 3 2020-02-22 22:28:25.046122
K 3 2020-02-22 22:28:29.602347 Key.navigation 2.5
....
```

Figure 1: File with the actions recorded

All analysis and measurements can be obtained from this actions file. Each line contains several information fields separated by tabs:

- The first field provides information about the kind of action. The actions are: START (task is started); PAUSE (task is paused); EXIT (user exits the application); RESTART (user restarts the task); IN (user enters into a segment); OUT (user exits a segment); K (keyboard action); M (mouse action); C (command action); B (user clicks a button on the application); F (application loses or gains focus); CLEAR (user clears all the content of the translation); RESTORE (user restores the content of the translation).

- The second field indicates the segment number.

- The third field gives the time and date of the event.

- Some actions have a fourth field which provides more detailed information about the event. For example, the key pressed, the text copied or pasted, and so on.

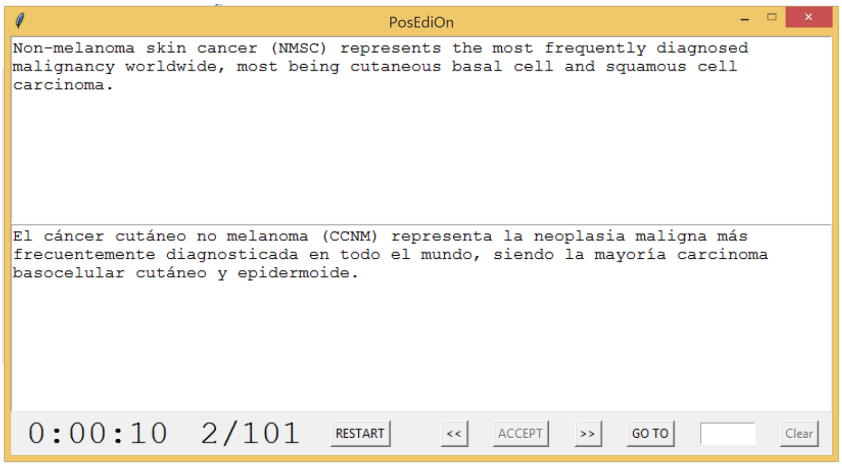

Figure 2: PosEdiOn interface

- Key actions have another field indicating the position in the target text where the key is pressed.

The user can pause and even stop the task and close the PosEdiOn program. Once the task is restarted, the new data will be appended to the existing actions file.

When the task is finished, the folder containing the program should be compressed again and sent back to the person who has to carry out the analysis.

3.2 User Interface

The interface displays the source and target language segments one on top of the other. Figure 2 shows the PosEdiOn interface, where the upper window contains the source segment and the lower window enables the translator to edit the text. Translators can see a wider context using the toolbar buttons located on the lower part, which can be used to move along the whole document.

Each unit is translated/edited one at a time and navigation through the different segments of the document can be achieved in four ways:

- Once the translator has finished post-editing a segment, he needs to validate it using the Ctrl+Enter keys. When this is done, the tool moves automatically to the next segment.
- To validate a segment, the user can also use the ACCEPT button. Once pressed, it also moves to the next segment.
- Using the << or >> buttons in the toolbar located at the lower part of the screen.
- Using the GO TO box, where you can write the number of the segment you want to move to.

Once a segment is accepted, its background turns green. The user can mark a segment as validated (green) using Ctrl+g; or he can change the state to undone (white background) using Ctrl+w. Segments can also be marked as red (Ctrl+r) to indicate a problematic status. Red segments can be reached directly using Ctrl+s.

3.3 Customization

In order to facilitate customization, certain elements can be modified in the *config.yaml* file without having to access the Python script.

As shown in Figure 4, users can customize the following elements:

- The size of the tool's window. Both height and width can be changed.
- Whether the source segment text can be edited or not. The edits introduced in the source segment are not registered by the tool. If the source segments can be edited, users can select and copy fragments of the source text.
- The size and type of font used for the source and target segments.
- Whether or not to show the chronometer.
- The name of the text file containing the task to translate or post-edit.
- The name of the actions file, where all the information containing the user's actions is stored.

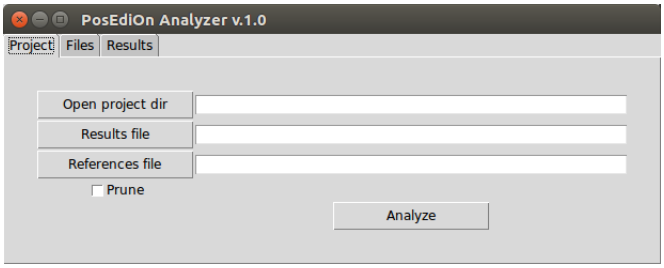

Figure 3: PosEdiOn analyzer interface

- The source and target language codes.
- The set of characters to be considered as symbols or punctuation. It also includes up to three user-defined groups of characters. In the example, a user-defined group called *mathematical* (containing symbols of mathematical operations) is defined.

```
Size:
    height: 10
    width: 80

Behaviour:
    allowEditSL: True

Font:
    font: courier 12

Chronometer:
    status: show
    #possible values: show / hide

Text:
    file: test-Google-1.txt

Actions:
    file: actions.txt

Languages:
    source: eng
    target: spa

Definition:
    symbols: "! @ # $ % ^ & ( ) _ { } [ ]"
    punctuation: ", : ; ."
    nameuserdef1: mathematical
    userdef1: "+ - * / ="
    nameuserdef2: None
    userdef2: None
    nameuserdef3: None
    userdef3: None
```

Figure 4: View of the customizable elements

4 PosEdiOn analyzer

PosEdiOn has a companion program, PosEdiOn analyzer, that performs different analyses on the PosEdiOn project files and offers a wide range of measurements. More specifically, it can calculate:

- Time spent editing each segment.
- HTER (Snover et al., 2006), the TER value comparing the raw MT output with the post-edited segment. A value of HTER is provided for each segment. The value of TER is calculated using tercom.[12]
- HBLEU, a BLEU (Papineni et al., 2002) value obtained comparing the raw MT output with the post-edited segment.
- HEd, an edit distance (Leveshtein distance) value calculated comparing the raw MT output with the post-edited segment.
- Keystrokes for each segment.

If a reference translation file is provided, the following measurements are also calculated:

- TER comparing the raw MT output with the reference translations. A value of TER is calculated for each segment.
- BLEU comparing the raw MT output with the reference translations. A value of BLEU is calculated for each segment.
- Ed, edit distance value calculated comparing the raw MT output with the post-edited segment.

To calculate the normalization of time, HEd (and eventually Ed) and keystrokes values, users can chose three different criteria: segment, token or character. All these values are provided both for each segment and for the whole document. On top of that, the mean and standard deviation are also calculated.

Users can choose to prune results. The pruning is based on a maximum value of normalized time and on a maximum value of normalized

[12]https://github.com/jhclark/tercom

segmentID	tokens	time	time$_norm$	HTER	HBLEU	HEd	HEd$_norm$	keys	keys$_norm$
1	5	38.02	7.6	0.1905	0.6703	18	3.6	18	3.6
2	11	48.81	4.44	0.12	0.6775	6	0.55	238	21.6
3	1	21.31	21.31	0.3333	0	8	8.0	10	10.0
4	29	279.69	9.64	0.2785	0.2318	72	2.48	148	5.1
5	15	72.12	4.81	0.0606	0.7242	2	0.13	50	3.3

Figure 5: Detailed information for each segment

keystrokes. These maximum values are calculated with the mean value and two times the standard deviation. All segments with a normalized time greater than the maximum or with a normalized number of keystrokes greater than the maximum are not taken into account to calculate the pruned values of all scores. The results are provided as numeric values and with a visual presentation of the results following the ideas of the Vis-Eval Metric Viewer (Steele and Specia, 2018).

4.1 Configuration

The configuration of the tool is performed using a Yaml configuration file (*config-analyzer.yaml*) as shown in Figure 6:

```
Filepath:
  path_in: /home/user/directory
  path_out: /home/user/directory
Results_file:
   prefix: results-
   sufix:
   extension: txt
Measures:
   bysegment: True
   normalization: tokens
   #one of segment, token, char
   HTER: True
   HBLEU: True
   HEd: True
   round_time: 2
   round_keys: 2
   round_HTER: 4
   round_HBLEU: 4
   round_HEd: 2
   round_other: 1
   TER: True
   BLEU: True
   ED: True
   round_TER: 4
   round_BLEU: 4
   round_Ed: 2
```

Figure 6: Yaml configuration file

The file paths including the location of the project and the results can be specified. The name of the results file can also be customized by adding a prefix, a suffix and an extension to the name of the project. If no prefix, suffix or extension is required, any of these fields can be left blank. The measurements can also be customized, and users can decide whether or not to show measurements by segment, the normalization criteria, which measurements will be calculated and shown, as well as the number of decimal points. Remember that the values of TER, BLEU and Ed will be calculated and shown only if a reference file is provided, regardless of the values in the configuration file.

4.2 Use of PosEdiOn analyzer

PosEdiOn analyzer can work both in text command and in graphical mode. To start the graphical user interface (shown in Figure 2) the program can be called with no parameters or with the `--gui` parameter. If no parameters or incomplete parameters are given, the GUI interface starts (see Figure 3). To use it in command line mode, you need to provide a set of parameters that can be checked using the `--h` option.

Usually we simply set the path for the directory containing the PosEdiOn project to analyze and the name of the output file containing the results:

```
python3 PosEdiOn-analyzer.py -p ./project
-o results.txt
```

If we want the results to be pruned, the option `--prune` should be used. Eventually we can set the name of a reference file containing the reference translation. The reference file is a text file that includes the reference translation aligned line by line with the text in the project.

PosEdiOn analyzer can also work with a set of files instead of a PosEdiOn project. This can be done using the Files tab, where the user can select the raw MT (option `--raw`), the post-edited files (`--ped`) and, optionally, the reference files (`--refs`) to calculate HTER, HBLEU and HEd values. If the reference files are provided, TER, BLEU and Ed values are also calculated. This allows PosEdiOn analyzer to be used independently from PosEdiOn tool.

4.3 Results

The analyzer can provide the following global results: time normalized, keystrokes normalized,

HTER, HBLEU and HEd. Remember that the normalization factor can be segment, token and character and can be set by the user. For each measurement, the mean and the standard deviation are provided. Pruned values are calculated rejecting the values lower than the mean minus two times the standard deviation or higher than the mean plus two times the standard deviation.

```
----------------------------------------
PRUNING:
----------------------------------------
time norm. mean: 9.19
time norm. std. dev.: 33.97
keys norm. mean: 6.36
keys norm. std. dev.: 28.25
max. norm. time: 77.14
max. norm. keystrokes: 62.86
----------------------------------------
IGNORED SEGMENT 9 norm.
time: 387.3 norm. kestrokes: 192.0
IGNORED SEGMENT 15 norm.
time: 212.24 norm. kestrokes: 301.0
IGNORED SEGMENT 19 norm.
time: 215.58 norm. kestrokes: 219.0
IGNORED SEGMENT 120 norm.
time: 122.75 norm. kestrokes: 3.5
IGNORED SEGMENT 189
norm. time: 67.42 norm. kestrokes: 75.0
----------------------------------------
TIME:
TIME TOTAL 19864.11
TIME NORM. MEAN 90.7
TIME NORM. STD 92.74

----------------------------------------
KEYS:
KEYS TOTAL: 12717
KEYS NORM MEAN 2.9
KEYS NORM STD 4.75
----------------------------------------
HTER:
HTER MEAN 0.1611
HTER STD 0.1172
----------------------------------------
HBLEU:
HBLEU MEAN 0.5303
HBLEU STD 0.2714
----------------------------------------
HEd NORM:
HEd NORM MEAN 1.28
HEd NORM STD 1.19
----------------------------------------
```

Figure 7: View of the results file

If the user has selected the detailed results through the *config-analyzer.yaml* file, the output file includes the following information for each segment (see Figure 5): segment ID, number of tokens or characters, time, time normalized, HTER, HBLEU, HEd, HEd normalized, number of pressed keys, number of pressed keys normalized.

PosEdiOn is able to generate graphics using the data, as the one shown in Figure 8 created from the pruned HTER values. The user can choose which data should be used to generate graphics.

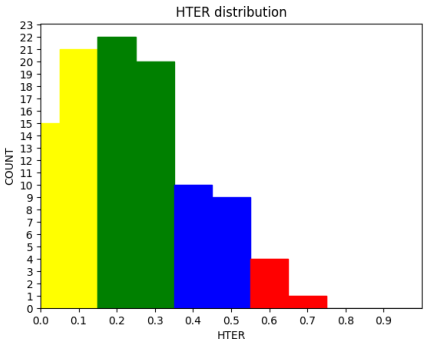

Figure 8: Graphic of the pruned HTER distribution

The results are stored in a tabulated text file, so they can be easily imported into any spreadsheet to perform further calculations.

5 Conclusions and future work

In this paper we have presented PosEdiOn, a tool to perform evaluations of post-editing tasks, and its companion program PosEdiOn analyzer, which allows to user to easily analyze the data obtained with PosEdiOn. Both programs are released under a free license (GNU GPL v3) and can be freely downloaded from the SourceForge page created for the project.[13]

We plan to use this tool in several studies related to post-editing and to implement new features such as the evaluation of fluency and adequacy and an error mark-up tool. Both programs are developed in Python3 and they can be easily adapted and improved. As the data are stored as tabbed text files, they can be easily processed or imported into any spreadsheet program to perform further analysis or data visualization.

Acknowledgements: The training of the neural machine translation systems used to develop and test PosEdiOn has been possible thanks to the NVIDIA GPU grant programme.

References

Allen, Jeffrey H. 2003. Post-editing. In Sommer, Harold, editor, *Computers and Translation: A translator's guide*. John Benjamin, Amsterdam.

[13] https://sourceforge.net/projects/posedion/

Aranberri, Nora, Gorka Labaka, Arantza Ilarraza, and Kepa Sarasola. 2014. Comparison of Post-Editing Productivity between Professional Translators and Lay Users. In *Proceedings of the Third Workshop on Post-Editing Technology and Practice (WPTP - 3)*, Vancouver, Canada.

Aziz, Wilker, Sheila C. M. De Sousa, and Lucia Specia. 2012. PET: a Tool for Post-editing and Assessing Machine Translation. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 3982–3987.

Barrault, Loïc, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 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, August. Association for Computational Linguistics.

Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Lmu Munich, Philipp Koehn, Jhu Edinburgh, Qun Liu, Varvara Logacheva, Mipt Moscow, Christof Monz, Matteo Negri, Matt Post, Johns Hopkins, Raphael Rubino, and Marco Turchi. 2017. Findings of the 2017 Conference on Machine Translation). In *Proceedings of the 2017 Conference on Machine Translation (WMT17))*, volume 2, pages 169–214.

Bojar, Ondřej, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 Conference on Machine Translation. In *Proceedings of the 2018 Conference on Machine Translation (WMT18)*, volume 2, pages 272–303.

Carl, Michael. 2012. Translog-II: A Program for Recording User Activity Data for Empirical Reading and Writing Research. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 4108–4112, Istanbul, Turkey, May. European Language Resources Association (ELRA).

Denkowski, M. and A. Lavie. 2012. TransCenter: Web-Based Translation Research Suite. In *AMTA 2012 Workshop on Post-Editing Technology and Practice Demo Session*, May.

Green, Spence, Jeffrey Heer, and Christopher D. Manning. 2013. The Efficacy of Human Post-editing for Language Translation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI 13*. ACM Press.

Guerberof, Ana. 2009. Productivity and Quality in MT Post-editing. In *Proceedings of Machine Translation Summit XII*.

Koponen, Maarit. 2016. Is Machine Translation Post-editing Worth the Effort? A Survey of Research into Post-editing and Effort. *The Journal of Specialised Translation*, pages 131–148.

Krings, Hans P. 2001. *Repairing texts: Empirical investigations of machine translation post-editing process*. The Kent State University Press, Kent, OH.

Moran, J., D. Lewis, and C. Saam. 2014. Analysis of Post-editing Data: A Productivity Field Test using an Instrumented CAT Tool. In O'Brien, Sharon, Laura Winther Balling, Michael Carl, Michel Simard, and Lucia Specia, editors, *Post-editing of Machine Translation: Processes and Applications*, chapter 6. Cambridge Scholars Publishing, United Kingdom.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wj Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, number July, pages 311–318.

Parra Escartín, Carla and Manuel Arcedillo. 2015. A Fuzzier Approach to Machine Translation Evaluation: A Pilot Study on Post-editing Productivity and Automated Metrics in Commercial Settings. In *Proceedings of the ACL 2015 Fourth Workshop on Hybrid Approaches to Translation (HyTra)*, volume 1, pages 40–45.

Plitt, Mirko and François Masselot. 2010. A Productivity Test of Statistical Machine Translation Post-Editing in a Typical Localisation Context. *The Prague Bulletin of Mathematical Linguistics NUMBER*, 93:7–16.

Shterionov, Dimitar, Riccardo Superbo, Pat Nagle, Laura Casanellas, Tony O'Dowd, and Andy Way. 2018. Human versus Automatic Quality Evaluation of NMT and PBSMT. *Machine Translation*, 32(3):217–235, Sep.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231, August.

Steele, David and Lucia Specia. 2018. Vis-eval Metric Viewer: A Visualisation Tool for Inspecting and Evaluating Metric Scores of Machine Translation Output. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 71–75.

Vieira, Lucas Nunes. 2013. An Evaluation of Tools for Post-editing Research: The Current Picture and Further Needs. In *Proceedings of the MT Summit XIV Workshop on Post-editing Technology and Practice (WPTP-2)*, pages 93–101, Nice.