# EASSE: Easier Automatic Sentence Simplification Evaluation

**Fernando Alva-Manchego**
University of Sheffield
f.alva@sheffield.ac.uk

**Louis Martin**
Facebook AI Research
Inria
louismartin@fb.com

**Carolina Scarton**
University of Sheffield
c.scarton@sheffield.ac.uk

**Lucia Specia**
Imperial College London
l.specia@imperial.ac.uk

## Abstract

We introduce EASSE, a Python package aiming to facilitate and standardise automatic evaluation and comparison of Sentence Simplification (SS) systems. EASSE provides a single access point to a broad range of evaluation resources: standard automatic metrics for assessing SS outputs (e.g. SARI), word-level accuracy scores for certain simplification transformations, reference-independent quality estimation features (e.g. compression ratio), and standard test data for SS evaluation (e.g. TurkCorpus). Finally, EASSE generates easy-to-visualise reports on the various metrics and features above and on how a particular SS output fares against reference simplifications. Through experiments, we show that these functionalities allow for better comparison and understanding of the performance of SS systems.

## 1 Introduction

Sentence Simplification (SS) consists of modifying the content and structure of a sentence to improve its readability while retaining its original meaning. For automatic evaluation of a simplification output, it is common practice to use machine translation (MT) metrics (e.g. BLEU (Papineni et al., 2002)), simplicity metrics (e.g. SARI (Xu et al., 2016)), and readability metrics (e.g. FKGL (Kincaid et al., 1975)).

Most of these metrics are available in individual code repositories, with particular software requirements that sometimes differ even in programming language (e.g. corpus-level SARI is implemented in Java, whilst sentence-level SARI is available in both Java and Python). Other metrics (e.g. SAMSA (Sulem et al., 2018b)) suffer from insufficient documentation or require executing multiple scripts with hard-coded paths, which prevents researchers from using them.

EASSE (Easier Automatic Sentence Simplification Evaluation) is a Python package that provides access to popular automatic metrics in SS evaluation and ready-to-use public datasets through a simple command-line interface. With this tool, we make the following contributions: (1) we provide popular automatic metrics in a single software package, (2) we supplement these metrics with word-level transformation analysis and reference-less Quality Estimation (QE) features, (3) we provide straightforward access to commonly used evaluation datasets, and (4) we generate a comprehensive HTML report for quantitative and qualitative evaluation of a SS system. We believe this package will facilitate evaluation and improve reproducibility of results in SS. EASSE is available in https://github.com/feralvam/easse.

## 2 Package Overview

### 2.1 Automatic Corpus-level Metrics

Although human judgements on grammaticality, meaning preservation and simplicity are considered the most reliable method for evaluating a SS system's output (Štajner et al., 2016), it is common practice to use automatic metrics. They are useful for either assessing systems at development stage, to compare different architectures, for model selection, or as part of a training policy. EASSE implementation works as a wrapper for the most common evaluation metrics in SS:

**BLEU** is a precision-oriented metric that relies on the proportion of n-gram matches between a system's output and reference(s). Previous work (Xu et al., 2016) has shown that BLEU correlates fairly well with human judgements of grammaticality and meaning preservation. EASSE uses

SacreBleu (Post, 2018)[1] to calculate BLEU. This package was designed to standardise the process by which BLEU is calculated: it only expects a detokenised system's output and the name of a test set. Furthermore, it ensures that the same preprocessing steps are used for the system output and reference sentences.

**SARI** measures how the simplicity of a sentence was improved based on the words added, deleted and kept by a system. The metric compares the system's output to multiple simplification references and the original sentence. SARI has shown positive correlation with human judgements of simplicity gain. We re-implement SARI's corpus-level version in Python (it was originally available in Java). In this version, for each operation ($ope \in \{add, del, keep\}$) and $n$-gram order, precision $p_{ope}(n)$, recall $r_{ope}(n)$ and F1 $f_{ope}(n)$ scores are calculated. These are then averaged over the $n$-gram order to get the overall operation F1 score $F_{ope}$:

$$f_{ope}(n) = \frac{2 \times p_{ope}(n) \times r_{ope}(n)}{p_{ope}(n) + r_{ope}(n)}$$

$$F_{ope} = \frac{1}{k} \sum_{n=[1,..,k]} f_{ope}(n)$$

Although Xu et al. (2016) indicate that only precision should be considered for the deletion operation, we follow the Java implementation that uses F1 score for all operations in corpus-level SARI.

**SAMSA** measures structural simplicity (i.e. sentence splitting). This is in contrast to SARI, which is designed to evaluate simplifications involving paraphrasing. EASSE re-factors the original SAMSA implementation[2] with some modifications: (1) an internal call to the TUPA parser (Hershcovich et al., 2017), which generates the semantic annotations for each original sentence; (2) a modified version of the monolingual word aligner (Sultan et al., 2014) that is compatible with Python 3, and uses Stanford CoreNLP (Manning et al., 2014)[3] through their official Python interface; and (3) a single function call to get a SAMSA score instead of running a series of scripts.

[1]https://github.com/mjpost/sacreBLEU

[2]https://github.com/eliorsulem/SAMSA

[3]https://stanfordnlp.github.io/stanfordnlp/corenlp_client.html

**FKGL** Readability metrics, such as Flesch-Kincaid Grade Level (FKGL), are commonly reported as measures of simplicity. They however only rely on average sentence lengths and number of syllables per word, so short sentences would get good scores even if they are ungrammatical, or do not preserve meaning (Wubben et al., 2012). Therefore, these scores should be interpreted with caution. EASSE re-implements FKGL by porting publicly available scripts[4] to Python 3 and fixing some edge case inconsistencies (e.g. newlines incorrectly counted as words or bugs with memoization).

## 2.2 Word-level Analysis and QE Features

**Word-level Transformation Analysis** EASSE includes algorithms to determine which specific text transformations a SS system performs more effectively. This is done based on word-level alignment and analysis.

Since there is no available simplification dataset with manual annotations of the transformations performed, we re-use the annotation algorithms from MASSAlign (Paetzold et al., 2017). Given a pair of sentences (e.g. original and system output), the algorithms use word alignments to identify deletions, movements, replacements and copies (see Fig. 1). This process is prone to some errors: when compared to manual labels produced by four annotators in 100 original-simplified pairs, the automatic algorithms achieved a micro-averaged F1 score of 0.61 (Alva-Manchego et al., 2017).

We generate two sets of automatic word-level annotations: (1) between the original sentences and their reference simplifications, and (2) between the original sentences and their automatic simplifications produced by a SS system. Considering (1) as reference labels, we calculate the F1 score of each transformation in (2) to estimate their correctness. When more than one reference simplification exists, we calculate the per-transformation F1 scores of the output against each reference, and then keep the highest one as the sentence-level score. The corpus-level scores are the average of sentence-level scores.

**Quality Estimation Features** Traditional automatic metrics used for SS rely on the existence and quality of references, and are often not enough to analyse the complex process of simplification. QE

[4]https://github.com/mmautner/readability

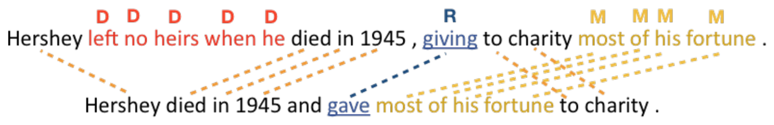

D D D D D
R
M M M M
Hershey left no heirs when he died in 1945 , giving to charity most of his fortune .
Hershey died in 1945 and gave most of his fortune to charity .

Figure 1: Example of automatic transformation annotations based on word alignments between an original (top) and a simplified (bottom) sentence. Unaligned words are DELETE. Words that are aligned to a different form are REPLACE. Aligned words without an explicit label are COPY. A word whose relative index in the original sentence changes in the simplified one is considered a MOVE.

leverages both the source sentence and the output simplification to provide additional information on specific behaviours of simplification systems which are not reflected in metrics such as SARI. EASSE uses QE features from Martin et al. (2018)'s open-source repository[5]. The QE features currently available are: the compression ratio of the simplification with respect to its source sentence, its Levenshtein similarity, the average number of sentence splits performed by the system, the proportion of exact matches (i.e. original sentences left untouched), average proportion of added words, deleted words, and lexical complexity score[6].

### 2.3 Access to Test Datasets

EASSE provides access to three publicly available datasets for automatic SS evaluation (Table 1): PWKP (Zhu et al., 2010), TurkCorpus (Xu et al., 2016), and HSplit (Sulem et al., 2018a). All of them consist of the data from the original datasets, which are sentences extracted from English Wikipedia (EW) articles. EASSE can also evaluate system's outputs in other custom datasets provided by the user.

**PWKP** Zhu et al. (2010) automatically aligned sentences in 65,133 EW articles to their corresponding versions in Simple EW (SEW). Since the latter is aimed at English learners, its articles are expected to contain fewer words and simpler grammar structures than those in their EW counterpart. The test set split of PWKP contains 100 sentences, with 1-to-1 and 1-to-N alignments (resp. 93 and 7 instances). The latter correspond to instances of sentence splitting. Since this dataset has only one reference for each original sentence, it is not ideal for calculating automatic metrics that rely on multiple references, such as SARI.

| Test Dataset | Instances | Alignment Type | References |
|---|---|---|---|
| PWKP | 93 | 1-to-1 | 1 |
| | 7 | 1-to-N | 1 |
| TurkCorpus | 359 | 1-to-1 | 8 |
| HSplit | 359 | 1-to-N | 4 |

Table 1: Test datasets available in EASSE. An instance corresponds to a source sentence with one or more possible references. Each reference can be composed of one or more sentences.

**TurkCorpus** Xu et al. (2016) asked crowdworkers to simplify 2,359 original sentences extracted from PWKP to collect multiple simplification references for each one. This dataset was then randomly split into tuning (2,000 instances) and test (359 instances) sets. The test set only contains 1-to-1 alignments, mostly with instances of paraphrasing and deletion. Each original sentence in TurkCorpus has 8 simplified references. As such, it is better suited for computing SARI and multi-reference BLEU scores.

**HSplit** Sulem et al. (2018a) recognised that existing EW-based datasets did not contain sufficient instances of sentence splitting. As such, they collected four reference simplifications of this transformation for all 359 original sentences in the TurkCorpus test set. Even though SAMSA's computation does not require access to references, this dataset can be used to compute an upperbound on the expected performance of SS systems that model this type of structural simplification.

### 2.4 HTML Report Generation

EASSE wraps all the aforementioned analyses in a simple comprehensive HTML report that can be generated with a single command. This report compares the system output with human reference(s) using simplification metrics and

[5]https://github.com/facebookresearch/text-simplification-evaluation

[6]The lexical complexity score of a simplified sentence is computed by taking the log-ranks of each word in the frequency table. The ranks are then aggregated by taking their third quartile.

QE features. It also plots the distribution of compression ratios or Levenshtein similarities between sources and simplifications over the test set. Moreover, the analysis is broken down by source sentence length in order to get insights on how the model handles short source sentence versus longer source sentences, e.g. *does the model keep short sentences unmodified more often than long sentences?* This report further facilitates qualitative analysis of system outputs by displaying source sentences with their respective simplifications. The modifications performed by the model are highlighted for faster and easier analysis. For visualisation, EASSE samples simplification instances to cover different behaviours of the systems. Instances that are sampled include simplifications with sentence splitting, simplifications that significantly modify the source sentence, output sentences with a high compression rate, those that display lexical simplifications, among others. Each of these aspects is illustrated with 10 instances. An example of the report can be viewed at `https://github.com/feralvam/easse/blob/master/demo/report.gif`.

## 3 Experiments

We collected publicly available outputs of several SS systems (Sec. 3.1) to evaluate their performance using the functionalities available in EASSE. In particular, we compare them using automatic metrics, and provide some insights on the reasoning behind their results (Sec. 3.2).

### 3.1 Sentence Simplification Systems

EASSE provides access to various SS system outputs that follow different approaches for the task. For instance, we include those that rely on phrase-based statistical MT, either by itself (e.g. PBSMT-R (Wubben et al., 2012)), or coupled with semantic analysis, (e.g. Hybrid (Narayan and Gardent, 2014)). We also include SBSMT-SARI (Xu et al., 2016), which relies on syntax-based statistical MT; DRESS-LS (Zhang and Lapata, 2017), a neural model using the standard encoder-decoder architecture with attention combined with reinforcement learning; and DMASS-DCSS (Zhao et al., 2018), the current state-of-the-art in the TurkCorpus, which is based on the Transformer architecture (Vaswani et al., 2017).

### 3.2 Comparison and Analysis of Scores

**Automatic Metrics** For illustration purposes, we compare systems' outputs using BLEU and SARI in TurkCorpus (with 8 manual simplification references), and SAMSA in HSplit. For calculating Reference values in Table 2, we sample one of the 8 human references for each instance as others have done (Zhang and Lapata, 2017).

When reporting SAMSA scores, we only use the first 70 sentences of TurkCorpus that also appear in HSplit.[7] This allows us to compute Reference scores for instances that contain structural simplifications (i.e. sentence splits). We calculate SAMSA scores for each of the four manual simplifications in HSplit, and choose the highest as an upper-bound Reference value. The results for all three metrics are shown in Table 2.

| | TurkCorpus | | HSplit |
|---|---|---|---|
| System | SARI | BLEU | SAMSA |
| Reference | 49.88 | 97.41 | 54.00 |
| PBSMT-R | 38.56 | **81.11** | **47.59** |
| Hybrid | 31.40 | 48.97 | 46.68 |
| SBSMT-SARI | 39.96 | 73.08 | 41.41 |
| DRESS-LS | 37.27 | 80.12 | 45.94 |
| DMASS-DCSS | **40.42** | 73.29 | 35.45 |

Table 2: Comparison of systems' performance based on automatic metrics.

DMASS-DCSS is the state-of-the-art in TurkCorpus according to SARI. However, it gets the lowest SAMSA score, and the third to last BLEU score. PBSMT-R is the best in terms of these two metrics. Finally, across all metrics, the Reference stills gets the highest values, with significant differences from the top performing systems.

**Word-level Transformations** In order to better understand the previous results, we use the word-level annotations of text transformations (Table 3). Since SARI was design to evaluate mainly paraphrasing transformations, the fact that SBSMT-SARI is the best at performing replacements and second place in copying explains its high SARI score. DMASS-DCSS is second best in replacements, while PBSMT-R (which achieved the highest BLEU score) is the best at copying. Hybrid is the best at performing deletions, but is the worst at replacements, which SARI mainly measures.

[7]At the time of this submission only a subset of 70 sentences had been released from HSplit. However, the full corpus will soon be available in EASSE.

The origin of the TurkCorpus set itself could explain some of these observations. According to Xu et al. (2016), the annotators in TurkCorpus were instructed to mainly produce paraphrases, i.e. mostly replacements with virtually no deletions. As such, copying words is also a significant transformation, so systems that are good at performing it better mimic the characteristics of the human simplifications in this dataset.

| System | Delete | Move | Replace | Copy |
|---|---|---|---|---|
| PBSMT-R | 34.18 | 2.64 | 23.65 | **93.50** |
| Hybrid | **49.46** | **7.37** | 1.03 | 70.73 |
| SBSMT-SARI | 28.42 | 1.26 | **37.21** | 92.89 |
| DRESS-LS | 40.31 | 1.43 | 12.62 | 86.76 |
| DMASS-DCSS | 38.03 | 5.10 | 34.79 | 86.70 |

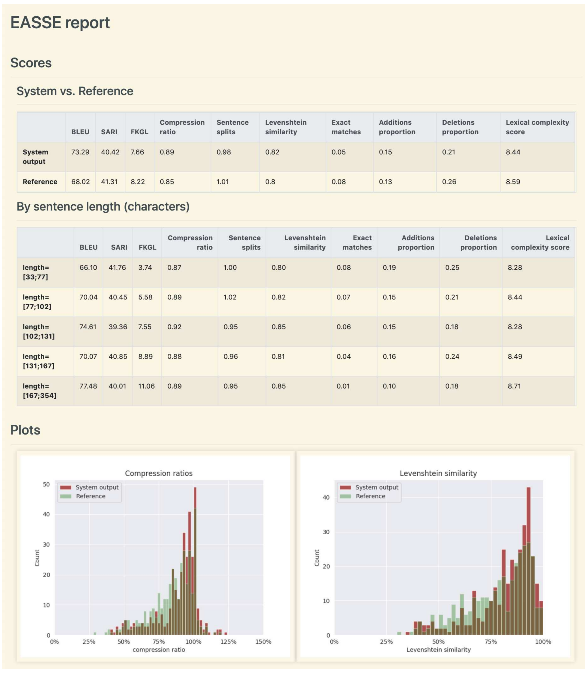
Table 3: Transformation-based performance of the sentence simplification systems in the TurkCorpus test set.

**Quality Estimation Features** Table 4 displays a subset of QE features that reveal other aspects of the simplification systems. For instance, the scores make it clear that Hybrid compresses the input way more than other systems (compression ratio of 0.57 vs. $\geq$0.78 for the other systems) but almost never adds new words (addition proportion of 0.01). This additional information explains the high Delete and low Replace performance of this system in Table 3. DRESS-LS keeps the source sentence unmodified 26% of the time, which does not show in the word-level analysis. This confirms that QE features are complementary to automatic metrics and word-level analysis.

| System | Compression ratio | Exact matches | Additions proportion | Deletion proportion |
|---|---|---|---|---|
| PBSMT-R | 0.95 | 0.1 | 0.1 | 0.11 |
| Hybrid | **0.57** | 0.03 | 0.01 | **0.41** |
| SBSMT-SARI | 0.94 | 0.11 | **0.16** | 0.13 |
| DRESS-LS | 0.78 | **0.26** | 0.04 | 0.26 |
| DMASS-DCSS | 0.89 | 0.05 | 0.15 | 0.21 |

Table 4: Quality estimation features, which give additional information on the output of different systems.

**Report** Figure 2 displays the quantitative part of the HTML report generated for the DMASS-DCSS system. The report compares the system to a reference human simplification. The "System vs. Reference" table and the two plots indicate that DMASS-DCSS closely matches different aspects of human simplifications, according to QE features. This contributes to explaining the high SARI score of the this system in Table 2.


EASSE report

Scores

System vs. Reference

| | BLEU | SARI | FKGL | Compression ratio | Sentence splits | Levenshtein similarity | Exact matches | Additions proportion | Deletions proportion | Lexical complexity score |
|---|---|---|---|---|---|---|---|---|---|---|
| **System output** | 73.29 | 40.42 | 7.66 | 0.89 | 0.98 | 0.82 | 0.05 | 0.15 | 0.21 | 8.44 |
| **Reference** | 68.02 | 41.31 | 8.22 | 0.85 | 1.01 | 0.8 | 0.08 | 0.13 | 0.26 | 8.59 |

By sentence length (characters)

| | BLEU | SARI | FKGL | Compression ratio | Sentence splits | Levenshtein similarity | Exact matches | Additions proportion | Deletions proportion | Lexical complexity score |
|---|---|---|---|---|---|---|---|---|---|---|
| **length=[33;77]** | 66.10 | 41.76 | 3.74 | 0.87 | 1.00 | 0.80 | 0.08 | 0.19 | 0.25 | 8.28 |
| **length=[77;102]** | 70.04 | 40.45 | 5.58 | 0.89 | 1.02 | 0.82 | 0.07 | 0.15 | 0.21 | 8.44 |
| **length=[102;131]** | 74.61 | 39.36 | 7.55 | 0.92 | 0.95 | 0.85 | 0.06 | 0.15 | 0.18 | 8.28 |
| **length=[131;167]** | 70.07 | 40.85 | 8.89 | 0.88 | 0.96 | 0.81 | 0.04 | 0.16 | 0.24 | 8.49 |
| **length=[167;354]** | 77.48 | 40.01 | 11.06 | 0.89 | 0.95 | 0.85 | 0.01 | 0.10 | 0.18 | 8.71 |

Plots

Figure 2: Overview of the HTML report for the DMASS-DCSS system (zoom in for more details).

## 4 Conclusion and Future Work

EASSE provides easy access to commonly used automatic metrics as well as to more detailed word-level transformation analysis and QE features which allows us to compare the quality of the generated outputs of different SS systems on public test datsets. We reported some experiments on the use of automatic metrics to obtain overall performance scores, followed by measurements of how effective the SS systems are at executing specific simplification transformations using word-level analysis and QE features. The former analysis provided insights about the simplification capabilities of each system, which help better explain the initial automatic scores.

In the future, we plan to continue developing the transformation-based analysis algorithms, so that more sophisticated transformations could be identified (e.g. splitting or subject-verb-object reordering). In addition, we expect to integrate more QE features to cover other aspects of the simplification process (e.g. depth of the dependency parse tree to measure syntactic complexity).

## References

Fernando Alva-Manchego, Joachim Bingel, Gustavo Paetzold, Carolina Scarton, and Lucia Specia. 2017. Learning how to simplify from explicit labeling of complex-simplified text pairs. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 295–305, Taipei, Taiwan. AFNLP.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2017. A transition-based directed acyclic graph parser for ucca. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1127–1138, Vancouver, Canada. ACL.

J.P. Kincaid, R.P. Fishburne, R.L. Rogers, and B.S. Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical Report 8-75, Chief of Naval Technical Training: Naval Air Station Memphis. 49 p.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. ACL.

Louis Martin, Samuel Humeau, Pierre-Emmanuel Mazaré, Éric de La Clergerie, Antoine Bordes, and Benoît Sagot. 2018. Reference-less quality estimation of text simplification systems. In *Proceedings of the 1st Workshop on Automatic Text Adaptation (ATA)*, pages 29–38, Tilburg, the Netherlands. ACL.

Shashi Narayan and Claire Gardent. 2014. Hybrid simplification using deep semantics and machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 435–445, Baltimore, Maryland. ACL.

Gustavo H. Paetzold, Fernando Alva-Manchego, and Lucia Specia. 2017. Massalign: Alignment and annotation of comparable documents. In *Proceedings of the IJCNLP 2017, System Demonstrations*, pages 1–4, Tapei, Taiwan. ACL.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Philadelphia, Pennsylvania. ACL.

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

Elior Sulem, Omri Abend, and Ari Rappoport. 2018a. Bleu is not suitable for the evaluation of text simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 738–744. ACL.

Elior Sulem, Omri Abend, and Ari Rappoport. 2018b. Semantic structural evaluation for text simplification. 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 685–696, New Orleans, Louisiana. ACL.

Md Sultan, Steven Bethard, and Tamara Sumner. 2014. Back to basics for monolingual alignment: Exploiting word similarity and contextual evidence. *Transactions of the Association for Computational Linguistics*, 2:219–230.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Sander Wubben, Antal van den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1015–1024, Jeju Island, Korea. ACL.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 595–605, Copenhagen, Denmark. ACL.

Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. 2018. Integrating transformer and paraphrase rules for sentence simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3164–3173, Brussels, Belgium. ACL.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 1353–1361, Stroudsburg, PA, USA. ACL.

Sanja Štajner, Maja Popović, Horacio Saggion, Lucia Specia, and Mark Fishel. 2016. Shared task on quality assessment for text simplification. In *Proceeding of the Workshop on Quality Assessment for Text Simplification - LREC 2016*, QATS 2016, pages 22–31, Portorož, Slovenia. ELRA.