A Multi-task Learning Approach to Text Simplification

Anna Dmitrieva$^{1,2}$ and Jörg Tiedemann$^1$

$^1$ University of Helsinki, Helsinki, Finland
{anna.dmitrieva,jorg.tiedemann}@helsinki.fi
$^2$ National Research University Higher School of Economics, Moscow, Russia

Abstract. We propose a multi-task learning approach to reducing text complexity which combines text summarization and simplification methods. For the purposes of this research, two datasets were used: the Simple English Wikipedia dataset for simplification and the CNN/DailyMail dataset for summarization.

We describe several experiments with reducing text complexity. One experiment consists in first training the model on summarization data, then fine-tuning it on simplification data. Another experiment involves training the model on both datasets simultaneously while augmenting source texts with a task-specific tag that shows the model which task (summarization or simplification) needs to be performed on a given text. Models with a similar architecture were also trained on each dataset separately for comparison. Our experiments have shown that the multi-task learning approach with task-specific tags is more effective than the fine-tuning approach, and the models trained for both tasks simultaneously can perform as good at each of them as the models that were trained only for that specific task.

Keywords: Text simplification · Text complexity · Abstractive summarization

1 Introduction

Text simplification is the process of reducing grammatical and lexical complexity of a text, while retaining its information content. In broad terms, text simplification can involve lexical and grammatical simplification, clarifying modification, when some concepts are explained in more detail, or summarization, when unnecessary or repeating fragments of text are being removed. Automatic text summarization, however, is a complex natural language processing task on its own. An automatic text summarization system generates a shorter version, or a summary, of a text which contains the most important information about a topic. A summary generated by an automatic text summarizer should consist of the most relevant information in a document and at the same time, it should occupy...
less space than the original document [5]. Text summarization can be extractive, when relevant sentences are simply extracted from the text, or abstractive, when the summary aims to capture the essence of the text, sometimes involving rewriting selected information.

These two tasks are sometimes combined in the development of systems for text complexity reduction. Saggion [18] lists multiple examples of such systems. In one of them, sentence simplification was used as a part of a multi-document extractive summarization algorithm. Sentences were simplified before clustering and selecting relevant sentences from each cluster [18]. In another system [14], lexical simplification is used during summary generation to replace difficult words. However, the usage of summarization techniques to aid simplification systems has not yet been explored. Considering summarization as a process that makes a text simpler, we propose a framework where summarization assists text simplification in a joint learning or a multi-task setup.

In this work, we propose to reduce text complexity using methods and datasets for text simplification and summarization. We aim to develop a system that in the future can be used for languages where datasets for training simplification models are not available. We believe that in most languages, summarization-specific datasets are easier to build than simplification datasets: for example by aligning Wikipedia article summaries (short paragraphs that preface the main body of text in an article) with the text of the article. Simplification-specific datasets, on the other hand, are difficult to build, since a lot of languages such as Russian do not have “simple” versions of regular media sources, like Simple Wiki or news/information written in simple language. Besides, both simplification and abstractive summarization require a good understanding of text semantics, so finding a proper way to preserve the meaning of texts will benefit both tasks.

2 Related Work

The field of automatic text simplification dates back to the middle of the 1990s [11]. Early work on automatic text simplification mainly aimed at grammar and style simplification in English. In the beginning, text simplification systems were mostly rule-based. One of the first works which dealt with automatic text simplification belongs to Chandrasekar and Srinivas [4]. Their research focused on syntax simplification techniques. The main goal was to speed up and reduce the error rate for other NLP tasks, such as syntactic parsing and machine translation.

Today the task of text simplification is often viewed as a monolingual translation problem [26]. Once viewed as such, a natural approach is to use sequence-to-sequence (seq2seq) neural machine translation (NMT) models [13]. The seq2seq NMT systems attempt to build a single large neural network such that every component is tuned based upon training sentence pairs [26]. While models of this kind tend to have drawbacks such as copying directly from the original sentence, making the output unnecessarily complex [13], there are solutions that can help to make up for these kinds of problems. Using deep neural networks
allows for creating simple and fluent modifications that preserve the meaning of
the original sentence [27]. State-of-the-art text simplification systems are able to
generate shorter and simpler sentences according to Flesch-Kincaid grade level
and human judgments, and provide a comprehensive analysis using human eval-
uation and a qualitative error analysis. Text simplification systems can also be
trained on unlabeled data in an unsupervised manner [23].

There are multiple approaches to text summarization. The majority of meth-
ods that are used today are methods of extractive summarization. However, in
recent years, with the rise of neural text generation techniques, abstractive tech-
niques are also becoming increasingly popular [16]. For example, in [6] a com-
pletely abstract model biased with a powerful content selector is proposed. In
this research, a data-efficient content selector is used to select phrases from a
source document that should be part of the summary. This approach not only
can be used for obtaining accurate summaries, but also can easily be transferred
to a new domain [6]. Finally, sometimes extractive and abstractive approaches
can be combined in one summarization system. For example, in [16] an extractive
summarization model is trained end-to-end using abstractive summaries.

All systems mentioned above in this section were trained on and work with
English language data. As mentioned before, simplification datasets are more
difficult to produce than summarization datasets, therefore for the majority of
languages the text simplification problem is understudied. Some examples of
non-English text simplification research include the creation of the Spanish sim-
plification corpus as part of the Simplext project [3] and developing a complex
system for extracting texts with high readability from a pre-crawled corpus and
performing text adaptation for learners of Russian as a foreign language [9].

3 Data

For the purposes of this research, two different datasets were used: one designed
for summarization and one for simplification. We used data aligned at the docu-
ment level instead of sentence-to-sentence alignment approach commonly used in
simplification. First, this alignment is common for text summarization datasets
as it allows models to learn to omit whole sentences and parts of the text. Sec-
ond, for languages without large simplification datasets, it would be easier and
less time-consuming to create a simplification dataset aligned at the document
level. Based on these prerequisites, the Simple Wikipedia dataset [10] was cho-
sen for simplification and the CNN/DailyMail dataset [15] for summarization.
Both of these datasets are frequently used for their respective tasks: for ex-
ample, CNN/DailyMail is used for training and testing summarization models in [6]
and [16] and Simple Wiki is used as part of a simplification dataset in [27]. Thus
we can conclude that, although these datasets have some drawbacks, they are
sufficient for correct training of our models.

Simplification. There are not many datasets dedicated specifically to the pur-
pose of text simplification. For approximately five years, Simple Wikipedia has
been the main source of data in this field [26]. Today, some new simplification datasets have emerged, including Newsela [26]. Although less noisy, they tend to provide only sentence-to-sentence alignments. That is why for the purposes of this research the Simple Wikipedia dataset was chosen. It comprises texts from Simple English Wikipedia and corresponding articles from English Wikipedia [10]. The data was downloaded from Wikipedia in May 2013 and comprises all Simple Wiki articles that at that time also had a corresponding article in English Wikipedia. This dataset has versions with sentence-to-sentence and document-to-document alignment. The document aligned version consists of full texts separated into sentences, each sentence provided with the article name and paragraph number. For the purposes of this research, we aligned the documents text-to-text, so the “regular” version of each text corresponded to its simpler version with the same name. This dataset has some flaws including noise and incorrect alignments (for instance, sometimes the last sentence of one article becomes the first sentence in another), but otherwise, it is the largest openly available dataset that suits the purpose of this research.

**Summarization.** For this task, we chose the CNN/DailyMail news articles dataset\(^1\). CNN/DailyMail comprises multi-sentence human-generated abstractive summaries of news articles aligned with full articles [15]. It is a version of the dataset that was presented in a shared task of question answering [7]. Articles were collected starting in April 2007 for CNN and June 2010 for the Daily Mail, both until the end of April 2015. Validation data is from March, test data is from April 2015 [7].

The parameters of the datasets used in this research are listed in Table 1.

|                  | CNN/DailyMail | Simple Wiki | All      |
|------------------|---------------|------------|----------|
| **Documents**    |               |            |          |
| Train set        | 287227        | 58775      | 346002   |
| Validation set   | 13368         | 500        | 13868    |
| Test set         | 11490         | 500        | 11990    |
| **Tokens (words)**|               |            |          |
| Train set source | 199570815     | 84975889   | 284546704|
| Train set target | 140506060     | 7867296    | 21917956 |
| Validation set source | 9092698 | 691540    | 9784238  |
| Validation set target | 73173 | 66381     | 139554   |
| Test set source | 7901203       | 726250     | 8627453  |
| Test set target | 597406        | 80049      | 677455   |

\(^1\) [https://github.com/harvardnlp/sent-summary](https://github.com/harvardnlp/sent-summary).
4 Preprocessing

Training: both datasets described above come with tokenization that inserts spaces between words and punctuation. For most of the experiments described here, that format was preserved. In addition, all texts were converted to lowercase and non-ASCII symbols were removed.

Sentence Boundary Tagging: For some of the models, an additional tokenization method was applied to the data. It has been shown that on the CNN/DailyMail dataset the summarization models can perform better if sentence boundary tagging is applied to the target text beforehand [6] like this: <t> w1 w2 w3 . </t>. We decided to try this technique on both datasets and compare the performance of the models.

Tagging Source Sentences with Task-specific Tags: In addition to training models separately on different datasets, we also attempted to train on both datasets simultaneously, merging them together and shuffling. We use a method introduced before for multilingual neural machine translation. In this method, an artificial token is added to the input sequence to indicate the required target language [8]. In our research, the artificial token in front of each text in the joined dataset indicates if the text needs to be summarized or simplified. This token is added in the front of the source sentence, so hereafter we call these tokens “front tags”. Our front tags are <2sum> for summarization and <2simp> for simplification.

 SentencePiece: As an experiment, another way of preprocessing using SentencePiece technology\(^2\) was tested on the Simple Wiki dataset. SentencePiece is a text tokenizer that implements a unigram language model and subword units like byte-pair encoding. For this experiment, all texts in the dataset were detokenized using the Moses toolbox\(^3\). After that, SentencePiece unigram and byte-pair encoding models were trained and used on the Simple Wiki texts. This tokenization technique was applied only on Simple Wiki and the joined datasets.

We use the OpenNMT-py toolkit [12] for preprocessing our data, as well as for training and testing the models. We follow the approach of See et al. [19] also used by Gehrmann et al. [6] for automatic summarization on CNN/DailyMail and truncate the source texts to 400 tokens and the target texts to 100 tokens. Although such truncation might seem brutal, in [19] it was proven that at least for the CNN/DailyMail dataset it can actually raise the performance of the summarization model. We also used a dynamic dictionary and shared vocabulary to ensure that source and target sentences are aligned and use the same dictionary, which is needed for copy attention (an implementation of pointer-generator networks that considers copying words from the source sequences [6,24]).

\(^2\) https://github.com/google/sentencepiece.
\(^3\) https://github.com/moses-smt/mosesdecoder.
Evaluation: Before evaluation, output texts produced by the models, as well as source and target test texts, were detokenized with the help of the Moses decoder or the SentencePiece decoder if the corresponding encoding was used. All additional tags were removed. Source and target texts were also cropped so that their length matches the size of the texts used for training: 400 words for source texts, 100 for target.

5 Models

For our experiments, we used the Pointer-Generator model architecture proposed in [19] and also used in [6]. The primary reason for the choice of the model and its parameters was twofold. First, the content selector used in this architecture has been proved to be simple yet accurate and improve the performance of summarization models [6]. Second, due to its data-efficiency, this technique can easily be adjusted to a new domain, which in our case is the simplification task [6]. Since this architecture has already been used on the CNN/DailyMail dataset, we are using similar model parameters to those used for this dataset before.

The model was built using the OpenNMT toolkit. It uses a one-layer LSTM with 512 hidden states and an embedding size of 128. The encoder is a bidirectional LSTM with 512 hidden states (256 in both directions). It uses copy-attention [24] which allows to copy words from the source. The attention mechanism that the model uses was introduced in [2]. Their approach allows for information to be spread throughout the sequence of annotations, which can be selectively retrieved by the decoder accordingly. The standard attention is then reused as copy attention. The loss is modified in a way that makes it divide the loss of a sequence by the number of tokens in it, which was proven by [6] to generate longer sequences during inference. This model uses Adagrad optimizer, no dropout, and gradient-clipping with a maximum norm of 2.

At the inference stage, beam search with a beam size of 10 is used, because it has been found out that bottom-up attention requires a larger beam [6]. Multiple penalties are applied: length penalty is used to encourage longer sequences, coverage penalty is used to avoid repetitions, and repeating trigrams are blocked.

The following setups were tried:

Experiments on One Dataset. In order to correctly evaluate the performance of the models trained on the two datasets, the models were also trained on each dataset separately. Moreover, separate models were trained on versions of the datasets with and without sentence boundary tagging. In addition to that, we also experimented with applying different SentencePiece tokenization and comparing the performance of models trained on data preprocessed with SentencePiece to that of models trained on regular data.

Experiments with Fine-tuning. In an attempt to train a model that would combine the summarization and simplification operations, we tried a fine-tuning
approach. We first trained a model on the summarization dataset and then fine-
tuned it on the smaller simplification dataset. For this experiment, we also tried
versions of the dataset with and without sentence boundary tagging.

**Experiments on the Joined Dataset.** As mentioned in the above paragraph,
some models were trained on both datasets simultaneously. All source texts were
augmented with task-specific tags. In one experiment, the model was trained on
a joined dataset with the volume of each original dataset preserved. In another
experiment, the summarization data was undersampled and the simplification
data was oversampled (with some texts being repeated) so that the amount
of source simplification data being 117,550 texts and the summarization data
being 287,227 texts. In the third experiment, the model mentioned in the first
experiment was additionally fine-tuned on simplification data that it had already
seen.

6 Evaluation

For evaluation we used metrics usually applied to text simplification and text
summarization evaluation. The metrics were BLEU, SARI and FKGL (Flesch-
Kincaid Grade Level) from the EASSE package [1] and also a pure Python
implementation of the ROUGE score\(^4\). Despite BLEU being widely used for dif-
ferent neural machine translation tasks, its effectiveness on text simplification
is sometimes doubted [22], that is why we also use SARI to evaluate the sim-
plification of texts. SARI measures how the simplicity of a sentence is improved
based on the words added, deleted and kept by a system [1]. For more extensive
evaluation of a text complexity reduction system it is indeed best to also have
human assessors, but in this initial study we limit ourselves to the listed metrics.

Since there has not been research featuring simplification or summarization
systems that were built using both datasets that we are using simultaneously,
or the Simple Wiki dataset aligned text-to-text the way that it is done here, it
is difficult to compare most of our results to other work. Therefore, for the most
part, we will only compare different results of our models.

**Experiments on One dataset.** Among the models trained on just one dataset,
be it CNN/DailyMail or Simple Wiki (not including the joined dataset), the best
ROUGE scores were obtained by the model trained on Simple Wiki data prepro-
cessed with SentencePiece BPE tokenization. However, in terms of BLEU, SARI
and FKGL scores the model trained on Simple Wiki without specific tokeniza-
tion seems to perform slightly better, outperforming both BPE and sentence
boundary tokenization (see Table 2).

In all tables below “R” stands for the ROUGE score. To make the data easier
to read, we limited ourselves to reporting only F-scores of ROUGE-1, 2 and L.

\(^4\) https://github.com/pltrdy/rouge.
Table 2. Testing the models trained on the Simple Wiki data on the Simple Wiki test set.

| Data type | BLEU  | SARI  | FKGL | R1 F | R2 F | RL F |
|-----------|-------|-------|------|------|------|------|
| BPE       | 22.89 | 51.11 | 7.77 | 0.46 | 0.31 | 0.47 |
| Tagged    | 22.97 | 50.65 | 7.67 | 0.50 | 0.26 | 0.41 |
| Plain     | 27.48 | 51.54 | 7.40 | 0.43 | 0.27 | 0.43 |

As for the models trained only on the CNN/DailyMail dataset, the version with sentence boundary tagging seems to give slightly better performance than the plain version in terms of simplicity and readability metrics (see Table 3). Nevertheless, the ROUGE scores of these models were not higher than those of some models described below (for example, compare Table 3 to Table 5).

Table 3. Testing the models trained on the CNN/DailyMail data on the CNN/DailyMail test set.

| Data type | BLEU  | SARI  | FKGL | R1 F | R2 F | RL F |
|-----------|-------|-------|------|------|------|------|
| Tagged    | 13.88 | 42.79 | 9.35 | 0.33 | 0.13 | 0.31 |
| Plain     | 13.44 | 42.70 | 9.33 | 0.33 | 0.13 | 0.31 |

Fine-tuning Experiments. The output of summarization models fine-tuned on simplification data (both with and without sentence boundary tagging), as seen in Table 4, proved to be inferior in quality to the output of models trained on one dataset. Readability, however, improved on the CNN/DailyMail test set. As can be seen, the presence of sentence boundary tagging does not make a big difference here, although it does slightly affect the performance. The ROUGE scores also were not improved in comparison to some other models (for example, compare Table 4 to Table 3 above):

Table 4. Testing the models trained on the CNN/DailyMail data and fine-tuned on the Simple Wiki data on the CNN/DailyMail test set.

| Data type                     | BLEU  | SARI  | FKGL | R1 F | R2 F | RL F |
|-------------------------------|-------|-------|------|------|------|------|
| CNNNDM fine-tuned on Simple Wiki | 10.53 | 41.01 | 8.91 | 0.32 | 0.12 | 0.30 |
| Tagged version                | 10.96 | 41.32 | 9.50 | 0.33 | 0.13 | 0.31 |

Experiments on the Joined Dataset. The models trained on the joined dataset with different configuration described above (see Sect. 5, Experiments on the joined dataset) obtained higher ROUGE scores on the CNN/DailyMail
test set compared to all other models. It should be noted that models with an architecture similar to ours trained on this dataset can get ROUGE-2 F-scores as big as 17.25 [6] (our highest score on this scale is 14), and the newest baseline ROUGE-2 F-score for CNN/DailyMail equals 19.24 [17].

| Data type                                      | BLEU    | SARI    | FKGL   | R1 F | R2 F | RL F |
|------------------------------------------------|---------|---------|--------|------|------|------|
| Joined dataset                                 | 12.38   | 41.52   | **10.08** | 0.34 | 0.14 | 0.32 |
| Undersample CNNDM, oversample SimpleWiki       | **13.02** | **41.93** | 10.21  | *0.35* | 0.14 | 0.32 |
| Joined dataset fine-tuned on SimpleWiki        | 12.50   | 41.59   | 10.20  | 0.34 | 0.14 | 0.32 |

However, the results shown on the Simple Wiki test set were less high than those described above in Table 2. Table 6 presents a comparison of three models: a model trained on the joined dataset, a model trained on undersampled summarization data and oversampled simplification data, and a model trained on joined dataset and fine-tuned on simplification data (see Sect. 5, Models). In terms of performance on the joined test set, the model trained on joined data without sampling or fine-tuning seems to have the best simplicity and readability scores. The ROUGE scores, however, did not really differ from one model to another.

| Data type                                      | BLEU    | SARI    | FKGL   | R1 F | R2 F | RL F |
|------------------------------------------------|---------|---------|--------|------|------|------|
| Joined dataset                                 | **14.10** | **42.51** | **9.79** | *0.35* | 0.14 | **0.33** |
| Undersample CNNDM, oversample SimpleWiki       | 11.63   | 39.86   | 9.94   | 0.33 | 0.13 | 0.31 |
| Joined dataset fine-tuned on SimpleWiki        | 13.61   | 42.05   | 10.21  | *0.35* | 0.14 | **0.33** |

The models generally perform better when evaluated on the test set from the same dataset they were trained on than on a test set from a different dataset. In our case, the results of models trained trained on the joined dataset are similarly high with the results of models trained on just one dataset when evaluated on a corresponding test set. For example, the scores obtained from the model trained on the joined data on the Simple Wiki test set were similar to, but not higher than those obtained on the same test set from the model trained specifically on Simple Wiki data (compare Table 7 to Table 2).

It should be noted that, unlike in models trained only on the Simple Wiki dataset, using SentencePiece tokenization did not improve the performance of the models trained on the joined dataset.
Table 7. Testing the models trained on the joined data on the Simple Wiki test set.

| Data type                                      | BLEU  | SARI  | FKGL | R1 F | R2 F | RL F |
|------------------------------------------------|-------|-------|------|------|------|------|
| Joined dataset                                 | 26.13 | 50.18 | 8.37 | 0.43 | 0.27 | 0.43 |
| Undersample CNNDM, oversample SimpleWiki       | 24.81 | 49.38 | 8.40 | 0.43 | 0.26 | 0.43 |
| Joined dataset fine-tuned on SimpleWiki        | 25.40 | 49.73 | 8.40 | 0.43 | 0.26 | 0.43 |

In order to see if the models understand the semantics of task-specific tags, we compared the performance of the models trained on the joined dataset on regular test sets and on test sets with reversed tags (where <2sum> becomes <2simp> and vice versa). Evaluation scores somewhat decreased but the difference was less than expected. The length of the output texts, however, has slightly increased on average, which is reflected in the increased readability scores (compare Table 8 to Table 6):

Table 8. Testing the models trained on the joined data on the joined test set with reversed front tags.

| Data type                                      | BLEU  | SARI  | FKGL | R1 F | R2 F | RL F |
|------------------------------------------------|-------|-------|------|------|------|------|
| Joined dataset                                 | 12.79 | 41.92 | 10.56| 0.35 | 0.14 | 0.33 |
| Undersample CNNDM, oversample SimpleWiki       | 11.58 | 38.79 | 10.09| 0.33 | 0.13 | 0.31 |
| Joined dataset fine-tuned on SimpleWiki        | 12.13 | 41.43 | 10.46| 0.35 | 0.14 | 0.33 |

A look at a small amount of randomly selected output texts confirmed that, on average, the simplified articles are longer than summaries. This is interesting given that all target texts are truncated to the same length during preprocessing. Examining the output also confirmed that the same source text with different tags will be processed differently by a model. This can be illustrated by the following example. Below is an article from the Simple Wiki dataset [10], which is still present in both English Wikipedia and Simple Wiki. These are the versions of the texts that were used for the evaluation of the models.

Original text: sofia wistam -lrb- born 15 may 1966 in liding, stockholm county, sweden -rrb- is a swedish television host on tv4 and tv3 and radio talk-show host. She has also worked as a stylist for stars such as carola, jerry williams and tommy nilsson. in 2008 she was also a judge on the talent show sweden’s got talent, during this year she also hosted her own radio show on rix fm. during 2009 sofia will host the competition show on swedish television.

Target text: sofia wistam -lrb- may 15 1966 -rrb- is a swedish television host and radio talk-show host.

Output with <2simp> tag: sofia wistam -lrb- born 15 may 1966 in liding, stockholm county, sweden -rrb- is a swedish television host on tv4 and tv3 and

5 https://en.wikipedia.org/wiki/Sofia_Wistam.
6 https://simple.wikipedia.org/wiki/Sofia_Wistam.
radio talk-show host. She has also worked as a stylist for stars such as carola, jerry williams and tommy nilsson.

**Output with <2sum> tag:** sofia wistam is a swedish television host on tv4 and tv3 and radio talk-show host. She has also worked as a stylist for stars such as carola, jerry williams and tommy nilsson.

The outputs with different tags are not the same even when the source text is quite short. However, it is hard to pinpoint the exact differences that each tag triggers, not only because the evaluation of such phenomena is generally difficult, but also because the tasks of summarization and simplification seem to be less distinguishable in their nature than, for example, the tasks of translating a text into two different languages.

7 Conclusion

In this paper we described the attempts to reduce text complexity using summarization and simplification data and models. We used different approaches such as fine-tuning summarization models on simplification data and training a model to solve both tasks simultaneously. In addition to that, different pre-processing techniques were applied. Our experiments have shown that, in terms of preprocessing, using the BPE tokenization seems to give better results in summarization for models trained on a single dataset. Sentence boundary tagging, although it improved summarization performance on the CNN/DailyMail dataset, did not seem to be effective on other data. As for training the models, using joined datasets with task-specific tags proved to be more effective than training a model for one task and fine-tuning it for another. The models trained on the joined dataset can perform different tasks with the same effectiveness as models trained for one task and tested on the corresponding data. The different tokenization approaches tested did not seem to have a significant effect on the performance of these models.

References

1. Alva-Manchego, F., Martin, L., Scarton, C., Specia, L.: EASSE: easier automatic sentence simplification evaluation. arXiv preprint arXiv:1908.04567 (2019)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
3. Bott, S., Saggion, H.: An unsupervised alignment algorithm for text simplification corpus construction. In: Proceedings of the Workshop on Monolingual Text-To-Text Generation, pp. 20–26 (2011)
4. Chandrasekar, R., Srinivas, B.: Automatic induction of rules for text simplification. Knowl.-Based Syst. 10(3), 183–190 (1997)
5. Gambhir, M., Gupta, V.: Recent automatic text summarization techniques: a survey. Artif. Intell. Rev. 47(1), 1–66 (2016). https://doi.org/10.1007/s10462-016-9475-9
6. Gehrmann, S., Deng, Y., Rush, A.M.: Bottom-up abstractive summarization. arXiv preprint arXiv:1808.10792 (2018)
7. Hermann, K.M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., Blunsom, P.: Teaching machines to read and comprehend. In: Advances in Neural Information Processing Systems, pp. 1693–1701 (2015)
8. Johnson, M., et al.: Enabling zero-shot translation: Google’s multilingual neural machine translation system. Trans. Assoc. Comput. Linguist. 5, 339–351 (2017)
9. Karpov, N., Sibirtseva, V.: Towards automatic text adaptation in Russian. Higher School of Economics Research Paper No. WP BRP, 16 (2014)
10. Kauchak, D.: Improving text simplification language modeling using unsimplified text data. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Long papers), vol. 1, pp. 1537–1546, August 2013
11. Keskisärkkä, R.: Automatic text simplification via synonym replacement (2012)
12. Klein, G., Kim, Y., Deng, Y., Senellart, J., Rush, A.M.: Opennmt: open-source toolkit for neural machine translation. arXiv preprint arXiv:1701.02810 (2017)
13. Kriz, R., et al.: Complexity-weighted loss and diverse reranking for sentence simplification. arXiv preprint arXiv:1904.02767 (2019)
14. Lal, P., Ruger, S.: Extract-based summarization with simplification. In: Proceedings of the ACL, July 2002
15. Nallapati, R., Zhou, B., Gulcehre, C., Xiang, B.: Abstractive text summarization using sequence-to-sequence RNNs and beyond. arXiv preprint arXiv:1602.06023 (2016)
16. Nallapati, R., Zhai, F., Zhou, B.: SummaRuNNer: a recurrent neural network based sequence model for extractive summarization of documents. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
17. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683 (2019)
18. Saggion, H.: Automatic text simplification. Synth. Lect. Hum. Lang. Technol. 10(1), 1–137 (2017)
19. See, A., Liu, P.J., Manning, C.D.: Get to the point: Summarization with pointer-generator networks. arXiv preprint arXiv:1704.04368 (2017)
20. Siddharthan, A.: A survey of research on text simplification. ITL-Int. J. Appl. Linguist. 165(2), 259–298 (2014)
21. Štajner, S., Saggion, H.: Data-driven text simplification. In: Proceedings of the 27th International Conference on Computational Linguistics: Tutorial Abstracts, pp. 19–23 (2018)
22. Sulem, E., Abend, O., Rappoport, A.: BLEU is not suitable for the evaluation of text simplification. arXiv preprint arXiv:1810.05995 (2018)
23. Surya, S., Mishra, A., Laha, A., Jain, P., Sankaranarayanan, K.: Unsupervised Neural Text Simplification. arXiv preprint arXiv:1810.07931 (2018)
24. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. In: Advances in Neural Information Processing Systems, pp. 2692–2700 (2015)
25. Wang, T., Chen, P., Rochford, J., Qiang, J.: Text simplification using neural machine translation. In: Thirtieth AAAI Conference on Artificial Intelligence (2016)
26. Xu, W., Callison-Burch, C., Napoles, C.: Problems in current text simplification research: new data can help. Trans. Assoc. Comput. Linguist. 3, 283–297 (2015)
27. Zhang, X., Lapata, M.: Sentence simplification with deep reinforcement learning. arXiv preprint arXiv:1703.10931 (2017)