Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference “Dialogue 2020”

Moscow, June 17–20, 2020

SENTENCE COMPRESSION FOR RUSSIAN: DATASET AND BASELINES

Kuvshinova T. (tatiana.kuvsh@yandex.ru)
Higher School of Economics, National Research University, Moscow, Russia

Sentence compression is the task of removing redundant information from a sentence while preserving its original meaning. In this paper, we approach deletion-based sentence compression for the Russian language. We use the data from the plagiarism detection corpus (ParaPlag) to create a corpus for sentence compression in Russian of almost 3,000 pairs of sentences. We align source sentences and their compressions using the Needleman-Wunsch algorithm and perform human-evaluation of the corpus by readability and informativeness.

Then we use bidirectional LSTM to solve sentence-compression task for Russian, which is a typical baseline for the problem. We also experiment with RuBert and Bert-multilingual. For the latter, we use transfer-learning, firstly pretraining the model on English data, which improves performance. We conduct human evaluation by readability and informativeness and do error analysis for the models. We are able to achieve f-measure of 74.8%, readability of 3.88 and informativeness of 3.47 (out of 5) on test data. We also implement post-hoc syntax-based evaluator, which can detect some of the wrong compressions, increasing overall quality of the system.

We provide the data and baseline results for future studies

Key words: sentence compression, summarisation, bert, rubert, lstm, corpus

DOI: 10.28995/2075-7182-2020-19-517-528

СЖАТИЕ ПРЕДЛОЖЕНИЙ РУССКОГО ЯЗЫКА: ДАТАСЕТ И БЕЙЗЛАЙНЫ

Кувшинова Т. (tatiana.kuvsh@yandex.ru)
Национальный исследовательский университет «Высшая школа экономики», Москва

Сжатие предложений — задача по удалению избыточной информации из предложения при сохранении его первоначального смысла. В этой статье мы обращаемся к сжатию предложений на основе удаления для русского языка. Мы используем данные из корпуса выявления плагиата (ParaPlag) для создания корпуса сжатых предложений русского языка, содержашего более чем 3000 пар предложений. Мы выравниваем исходные предложения и их сжатия, используя алгоритм Нидлмана-Вунша, и проводим ручную оценку корпуса по читаемости и информативности.

Затем мы используем двунаправленную LSTM для решения задачи сжатия предложений русского языка, что является типичным способом решения этой задачи. Мы также экспериментируем с RuBert и многоязычным Bert. В последнем случае мы используем трансферное обучение, сначала обучая модель на английских данных, что улучшает качество работы системы. Мы проводим ручную оценку по читаемости и информативности и анализ ошибок для моделей. Мы достигли f-меры 74,8%, читаемости 3,88 и информативности 3,47 (из 5) на тестовых данных. Кроме того, мы разработали синтаксический оценщик, который может распознать некоторые из неправильных сжатий предложений, позволяя увеличить общее качество системы компресии.

Мы предоставляем данные и результаты бейзлайнов для будущих исследований.

Ключевые слова: сжатие предложений, сокращения, суммаризация, корпус

1. Introduction

Sentence compression is a text-to-text rewriting task of shortening a sentence by omitting redundant information while preserving the main points. Usually, a “perfect” compression is a difficult paraphrase task, involving removing, reordering and inserting operations. However, the problem is often narrowed down to deletion-based sentence compression.

Deletion-based sentence compression is performed by removing some words from the original sentence. The remaining words form compression in the exact same order as they appeared in the original sentence. This problem can be solved as a sequence labeling binary classification task—we classify each token in the original sentence with “stay” or “delete” label and then remove all the tokens with the “delete” label.

Sentence compression for English has attracted many researchers. An important milestone in the field was the release of a large parallel compression corpus by Google [4] and as well as applying deep learning techniques, especially LSTMs to the problem [3]. The most recent results in the field are achieved by complex models, combining LSTM with some additional techniques, for example, language model evaluator [11], [18].

However, to the best of our knowledge sentence compression has never been solved for Russian, which is easily explained by the lack of parallel data and complex structure of the language. In this paper, we approach deletion-based sentence compression for Russian. We present a parallel sentence-compression corpus for Russian with almost 3 thousand sentences based on The ParaPlag [15] data. We also conduct experiments with several data-driven models for sentence compression, which are bidirectional LSTM, RuBert-compression, and MultiBert-compression. We build syntax-tree based evaluator over compression systems to improve compression quality and perform an automatic and human evaluation of the results.

We hope that our work will serve as a pivot point for future researches.

2. Data

Data-driven solutions to natural language processing problems are usually data-hungry. Deletion-based sentence compression requires a large parallel corpus with sentence-compression pairs. Such data has no evident natural source (as, for example, parallel translation data) and no specific sentence compression corpora have yet been released for Russian.

In this paper, we use data from the Russian dataset for paraphrased plagiarism detection [15] to construct a deletion-based sentence-compression corpus for Russian. The corpus contains about 20 thousand manually paraphrased source-plagiarism pairs. Sometimes, when a person plagiarises a sentence, he or she summarizes or expands it. Fortunately, such examples are annotated in the corpus. We use them as a source for our dataset.

There are 7,298 such pairs of sentences in the dataset. Some of them are also paraphrased, in such cases we resolve the paraphrase before further processing. For example, for the source pair of sentences:

> *Sentence: Книги Толкина послужили основой для создания множества настольных, компьютерных и видеоигр как для PC, так Mobile.*
>
> *Compression: Книги Толкина стали основой для создания настольных, компьютерных и видеоигр.*

As a result of paraphrase resolution the compression will become:

> *Книги Толкина послужили основой для создания настольных, компьютерных и видеоигр.*

We exclude those sentences where a paraphrase cannot be resolved automatically (for example when it is adjacent to deletion and its boundaries cannot be clearly established).

Then we tokenize a source sentence and its compression and align them using the Needleman-Wunsch algorithm for sequence alignment [13]: a dynamic programming approach firstly developed for DNA sequence alignment. The result of the alignment for the example sentence given above will be:

Finally, we exclude all the pairs with a compression rate higher than 0.7 (traditionally, small compression rate corresponds to shorter compressions, i.e. a compression rate is a proportion of words to stay). We make that decision because the sentences with high compression rate form poor training material: they are only partly compressed, some words that *could be* removed are not removed from them. Thus, a compression system would be confused by the training data inconsistency.

The remaining data contains 2,955 pairs of sentences. We split it into the train and test set as follows:

Table 1: The parallel sentence-compression corpus statistics and train-test split

set	no. sentences	no. tokens	compression rate
train	2,659	66,238	0.55
test	296	7,450	0.56

Common metrics for compression quality evaluation are readability and informativeness [12]. Readability addresses grammatical correctness and naturality of a compressed sentence. Informativeness shows a degree to which original meaning was preserved in the compression. We asked three native Russian speakers to annotate the first 150 sentence-compression pairs from the test set by readability and informativeness using a 5-grade scale. The first annotator has annotated all 150 pairs of sentences, the second—the first 100 pairs and the third—the pairs from 101 to 150, thus giving two annotations for every pair of sentences. As a result, we got a mean readability of 4.68 and mean informativeness of 4.03.

We make the sentence-compression corpus available online for future researches.[1]

3. Experiments

3.1. LSTM

Bidirectional long-short term memory networks form strong baselines and are used as baselines in the majority of recent papers about sentence compression in English [3, 7, 11, 18]. That is why we decided to use biLSTM as our baseline as well.

We build a 3-layered biLSTM network with input and hidden layers of 300 dimensions and a *softmax* output layer. We use binary cross-entropy loss function and Adam optimizer. We initialise our model with *tayga_none_fasttextcbow_300_10_2019* vectors by RusVectōrēs project [10]. We also allow the model to finetune embeddings

[1] download at https://goo-gl.su/EqQvj8

during training. We lemmatize the corpus before training making it compatible with embeddings using the Russian UdPipe Syntagrus model [16].

We implement our model using PyTorch and train it on sentence-compression data for Russian with the following parameters: learning rate of $1e-3$, weight decay of $1e-3$, dropout of 0.35 and mini-batch size of 20 sentences. We do not finetune any parameters and train the model until test loss starts increasing, which is 20 epochs on our data.

We then evaluate the model and present the results in the **Evaluation** section.

3.2. RuBert

BERT model has outperformed state of the art results in many natural language processing tasks [1], so it was only natural to approach sentence compression task with it.

We add a linear classification layer on top of the BERT hidden-states output and use Cross-Entropy for the loss function. We use Adam optimizer.

We implement the model using PyTorch transformers and use the pretrained RuBert model by DeepPavlov [8] to initialize the model. We convert Deeppavlov RuBert checkpoint to PyTorch-transformers Bert checkpoint to be able to use the same architecture and training procedure for RuBert and original Multilingual Bert model (described in the next subsection). Then we train it on Russian compression data with a learning rate of $3e-5$ and minibatch of 8 sentences until the loss starts increasing on the test set, which takes 6 epochs for our data. We do not finetune any parameters.

The evaluation results are presented in the **Evaluation** section.

3.3. M-Bert

Multilingual Bert was proven to be good at cross-lingual model transfer [14], i.e. finetuning M-Bert to solve a task on a one language data allows solving the same task for other languages with little drop in quality. Authors hypothesize that the reason behind that behavior may be that similar words in different languages are surrounded by the same universal symbols (such as punctuations, numbers, etc.) and therefore learn similar vector representation. Therefore it makes sense to finetune M-Bert on English sentence-compression data before solving the task for Russian.

We use the same model architecture as for RuBert model. Firstly, we finetune the model on 8,000 English sentence-compression pairs from the first release of the Google sentence-compression dataset [4]. We train the model until the loss of 1,000 English test set starts increasing, which takes 3 epochs. Then we further train it on Russian sentence-compression pairs, again, until an increase in test loss, which takes 3 epochs.

For comparison we also train M-Bert model on Russian data only with the same parameters (i.e. skipping the first part of training procedure). This enables us to distinguish between the impact of fine-tuning and initial model quality.

Finally, we evaluate the models and present the results in the **Evaluation** section.

4. Post-hoc syntax evaluator

4.1. Foundation

Compression models performances can be inconsistent: they can compress correctly one sentence and fail with another. This sort of behaviour is highly undesirable for a service and neural network algorithms are hardly controlled and debugged. That is why we considered building post-hoc evaluator to separate correctly compressed sentences from damaged ones.

Inspired by tree cutting approaches to sentence compression [5], [6], [9], we decided to build a syntax-tree based evaluator. The tree cutting approach is based on the idea that parts of a sentence close to dependency graph root are preserved during compression, while leaves are dropped out. At the same time it is ungrammatical to remove a head preserving it's child, i.e. the resulting graph ought to be a subgraph of the source graph with the same root.

Consider the following example: *Сокращая предложения, будьте очень внимательны.* and its dependency graph below, as well as dependency graphs of its deletion-based compressions.

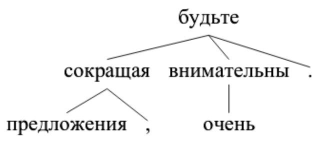

Figure 1: Source sentence dependency graph

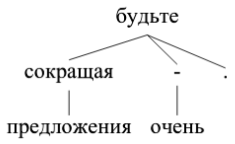

Figure 2: Correct compression graph

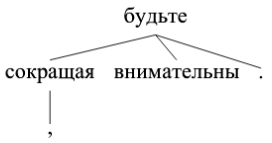

Figure 3: Correct compression graph

We see that potentially correct compression *Сокращая, будьте внимательны.* is a subgraph of the source sentence graph. On the opposite, potentially wrong compression has a "cut off" node *очень*, which has lost it's head. Such transformation is usually ungrammatical and should not be allowed in the compression system.

4.2. Implementation

We use Universal Dependencies syntax parser [17] and Russian Udpipe Syntagrus model [2] for syntax parsing. Firstly, we build syntax trees for source and compressed sentences. Then we count tree distance—number of nodes that have changed their parents (which presumably happened because their original head was removed by compression system).

Mostly, any deviation from a source sentence subgraph indicates an ungrammatical compression. However, due to possible syntactic parser mistakes and several corner cases such as conjunction treatment, we would like to allow at least some fluency to an algorithm. To determine a threshold after which we consider a compression wrong, we calculate tree-distances between source sentences and gold compressions from the train dataset. We normalize distances over compression length to get a proportion of tokens that changed their heads and get the following normalized tree distance distribution:

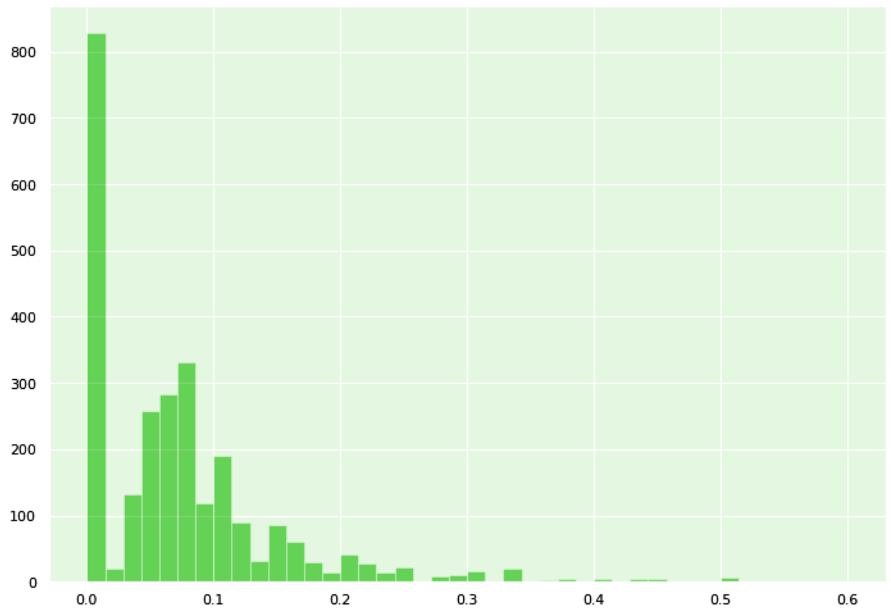

Figure 4: Normalized tree distance distribution over the train set

Based on the distribution and examples analysis we chose a threshold of 0.1—all compressions with higher proportion of misplaced nodes are considered wrong.

Our syntax evaluator doesn't directly improve the models performances, it only shows probably ungrammatical compressions, cases where the model have failed. Leaving those sentences uncompressed allows to trade off recall for precision.

5. Evaluation

While automatic evaluation using test data is a good way to access model performance, it is not sufficient for sentence compression task. For example, removing a subject from a sentence can give us accuracy of $> 90\%$, and yet the sentence can become completely ungrammatical and uninformative. It is possible that in an experiment a model with the maximum F-score and a model producing the most readable compressions are different ones [11]. That is why we perform both automatic and human evaluation of the results.

5.1. Automatic evaluation

We evaluated the models' performance on our 296 sentences test-set. For every model we evaluated its performance as a standalone system and combined with syntax evaluator. When using syntax evaluator, we excluded those compressions that were marked as probably wrong, i.e. evaluated only sentences approved by syntax evaluator. You can see evaluation results in the table below:

Table 2: F1-score on test data for LSTM, RuBert and M-Bert models; "f1-delete" is F1 measure for delete labels, "f1-stay" is F1 measure for stay labels and "f1" is weighted-averaged F1 measure between labels; SE stands for Syntax evaluator

model	f1, %	f1-delete, %	f1-stay, %	training data
LSTM	74.8	71.3	77.5	2,659 Russian pairs
LSTM + SE	77.1	70.1	81.3	2,659 Russian pairs
RuBert	67.9	63.0	71.6	2,659 Russian pairs
RuBert + SE	70.2	68.5	71.7	2,659 Russian pairs
M-Bert (ru)	67.1	63.8	72.1	2,659 Russian pairs
M-Bert (ru) + SE	68.2	64.9	73.5	2,659 Russian pairs
M-Bert	69.3	64.4	73.0	8,000 English + 2,659 Russian pairs
M-Bert + SE	70.8	65.5	74.5	8,000 English + 2,659 Russian pairs

As we can see, baseline LSTM surprisingly performs better than Bert-base models. M-Bert trained with both English and Russian data exceeds both Ru-Bert and M-Bert trained with Russian data only, it shows the promise of using transfer-models for sentence compression task.

Systems using syntax evaluator show higher performances than row models, however they leave some sentences uncompressed.

5.2. Human evaluation

We asked three native Russian speakers to evaluate systems' compressions as well as gold compressions of the same sentences by readability and informativeness using a 5-grade scale. The first annotator has evaluated compressions for the first 150 sentences of the test-set (600 sentences), the second—the first 100 sentences of the test-set (400 sentences) and the third—compressions of sentences from 101 to 150 of test-set (200 sentences). Compressions were presented to annotators in random order so that they don't know which system (or gold compression) they are evaluating. Agreement rate (Pearson correlation) between annotators is presented in the table below:

Table 3: Agreement rate (Pearson correlation) between annotators; the p-value is less than $1e - 9$ for every test

annotator pair	readability	informativeness	sentences in common
1 & 2	0.75	0.66	1–100
1 & 3	0.86	0.78	101–150

We consider the agreement rate satisfactory.

We also evaluate test data after processing it with syntax evaluator. We exclude sentences and compressions marked by syntax evaluator as inappropriate before calculating metrics. Note, that this inevitably increases compression rate, as some sentences remain uncompressed. We evaluate gold compressions by the same procedure.

Table 4: Mean readability, informativeness and compression rate for LSTM, RuBert, M-Bert models and gold data for the first 150 sentences of the test set; SE stands for syntax evaluator; higher compression rate corresponds to longer compressions and compression rate includes uncompressed sentences for models with SE

model	readability	informativeness	sentences excluded by SE	compression rate
Gold	4.68	4.03	0	0.56
Gold + SE	4.73	4.13	35	0.73
LSTM	3.84	3.29	0	0.54
LSTM + SE	4.10	3.61	49	0.78
RuBert	3.46	3.26	0	0.58
RuBert + SE	3.59	3.42	81	0.85
M-Bert	3.88	3.47	0	0.62
M-Bert + SE	4.07	3.71	55	0.84

We provide evaluation results for gold compressions from the test set for better comparison. We encourage any future researches of sentence compression in Russian to evaluate gold compressions as well. The reason is different annotators may have different ideas about subjective metrics as readability and informativeness, so human-evaluation results are hardly comparable between studies. However, evaluating the same set of gold compressions will allow leveling systems' results *in comparison* with gold compressions.

We see that M-Bert is better than other models in both readability and informativeness, even though it had had much lower f-measure than the LSTM model. We see that apparently training on a large English corpus allows the model to better generalize in sentence-compression task.

Applying syntax evaluator after compression system (excluding syntactically broken compressions) increases compressions readability and informativeness at the expense of increasing compression rate. The difference between a raw model and a model with post hoc evaluator is higher for initially worse models (where more compressions had to be excluded). However, RuBert model shows relatively low readability and informativeness even after syntax evaluator excluded more than a half of its compressions from analysis. After applying syntax evaluator the systems remove only about 15–20% of words on average, but this may be desirable results in some cases.

5.3. Examples and error analysis

While we consider evaluation results more than satisfactory for the first experience in sentence compression for Russian, there are still many cases when the models fail to produce grammatical and informative compression. We will discuss them and possible approaches to improving systems' performance in this subsection. We mark those compressions which were considered inappropriate by syntax evaluator with asterisk (*).

One of the frequent mistakes made by the models is the meaning travesty. Words deletion can sometimes radically change meaning as in the following examples:

(1)	**Source sentence:**	*К тому времени Фрида уже не могла встать с постели.*
	LSTM:	*Фрида не могла встать.*
	RuBert:	*Фрида могла встать с постели.*
	M-Bert:	*Фрида уже не могла встать с постели.*
(2)	**Source sentence:**	*В октябре 2011 года робот CubeStormer II, специально собранный из 4 наборов конструктора Lego Mindstorms, побил рекорд человека и собрал кубик за 5,53 секунды.*
	LSTM:	*В октябре 2011 года робот CubeStormer II, специально собранный из 4 наборов конструктора Lego Mindstorms, побил рекорд человека.*
	RuBert:	*В октябре 2011 года робот CubeStormer II, специально собранный из 4 наборов Lego Mindstorms побил.*
	*** M-Bert:**	*В октябре 2011 года робот CubeStormer II, специально собранный из 4 наборов конструктора Lego Mindstorms, побил человека.*

In the first example, RuBert system removes negation giving the sentence an opposite meaning. The syntax evaluator does not penalise those compressions, because in Universal Dependencies grammar negation is dependent on its verb. This case can be, however, addressed by merging negation to the following verb at preprocessing stage (which, of course, will require embeddings trained in such way).

In the second example M-Bert system removes word *рекорд*, the sentence stays completely grammatical, but the meaning changes from *The robot broke human record* to *The robot beat a man*... The head in noun phrase *рекорд человека* is *рекорд*. The complement *человека* cannot stay when its head was removed, but the system is apparently confusing the noun phrase for a typical *Adjective+Noun* phrase and is removing the first word. Providing the models with some syntactic information could improve performance. At the same time, in this case syntax evaluator is able to detect wrong transformation and marks the compression as inappropriate.

Another typical situation when the systems produce wrong compressions is the presence of a point inside a sentence (in an abbreviation or as a part of a number). It would appear that the system mistake this point to end-of-sentence point and get rather confused seeing tokens passed after it. The resulting compressions are barely readable:

(3)	**Source sentence:**	*В финале соревнования по игре Painkiller в 2005 г. его выигрыш составил, ни много ни мало, 150 000 долларов.*
	*** LSTM:**	*В финале соревнования по игре.*
	*** RuBert:**	*В финале соревнования по игре Painkiller. 150 000 долларов.*
	*** M-Bert:**	*В финале соревнования по игре Painkiller в 2005 г. его выигрыш составил ни.*

To avoid such mistakes it is possible to remove inline points or change them to different symbols while preprocessing. Syntax evaluator detects severe changes in syntax structure in all systems' compressions.

6. Results

In this paper, we have addressed the problem of deletion-based sentence compression in Russian. We have prepared and preprocessed a parallel corpus of 2,955 sentence-compression pairs based on the data if ParaPlag corpus [15]. We asked three native Russian speakers to evaluate the corpus quality and got mean scores of 4.68 for readability and 4.03 of informativeness (out of 5).

We then build three sentence compression systems for Russian: bidirectional LSTM, RuBert based model and multilingual Bert based model. We train LSTM and RuBert with our corpus and multilingual Bert on both English sentence-compression data from [4] and our corpus.

We also implement post-hoc syntax evaluator based on source and compression syntax trees comparison. The evaluator does not improve compressions, but is able to detect some of wrong compressions, which could prove useful in applying systems to real data.

We conduct both automatic and human evaluation of the results. LSTM model achieves the highest f-measure on the test set, which is 74.8%. The multilingual Bert based model is the best judging by human evaluation: it achieves readability of 3.88 and informativeness of 3.47 (out of 5).

Applying syntax evaluator (evaluating only compressions approved by syntax evaluator) allows to achieve f-measure of 81.3% (for LSTM model), readability of 4.10 and informativeness of 3.71 (for multilingual Bert). However, the system with syntax evaluator simply doesn't compress lots of sentences, increasing average compression rate as the result.

We give several examples of models' performance and do error analysis.

Our work is the first to the best of our knowledge dedicated to sentence compression in Russian. We consider the achieved results as a good start and certainly hope that future researches will consider the topic and will be able to surpass our results.

References

1. *Devlin, J. et al.:* Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. (2018).
2. *Droganova, K. et al.:* Data conversion and consistency of monolingual corpora: Russian ud treebanks. In: Proceedings of the 17th international workshop on treebanks and linguistic theories (tlt 2018), december 13–14, 2018, oslo university, norway. pp. 52–65 Linköping University Electronic Press (2018).
3. *Filippova, K. et al.:* Sentence compression by deletion with lstms. In: Proceedings of the 2015 conference on empirical methods in natural language processing. pp. 360–368 (2015).
4. *Filippova, K., Altun, Y.:* Overcoming the lack of parallel data in sentence compression. In: Proceedings of the 2013 conference on empirical methods in natural language processing. pp. 1481–1491 (2013).
5. *Filippova, K., Strube, M.:* Dependency tree based sentence compression. In: Proceedings of the fifth international natural language generation conference. pp. 25–32 Association for Computational Linguistics (2008).
6. *Filippova, K., Strube, M.:* Sentence fusion via dependency graph compression. In: Proceedings of the conference on empirical methods in natural language processing. pp. 177–185 Association for Computational Linguistics (2008).
7. *Klerke, S. et al.:* Improving sentence compression by learning to predict gaze. arXiv preprint arXiv:1604.03357. (2016).
8. *Kuratov, Y., Arkhipov, M.:* Adaptation of deep bidirectional multilingual transformers for russian language. arXiv preprint arXiv:1905.07213. (2019).
9. *Kurisinkel, L. J. et al.:* Domain adaptive neural sentence compression by tree cutting. In: European conference on information retrieval. pp. 475–488 Springer (2019).
10. *Kutuzov, A., Kuzmenko, E.:* WebVectors: A toolkit for building web interfaces for vector semantic models. In: Ignatov, D. I. et al. (eds.) Analysis of images, social networks and texts: 5th international conference, aist 2016, yekaterinburg, russia, april 7–9, 2016, revised selected papers. pp. 155–161 Springer International Publishing, Cham (2017).
11. *Kuvshinova, T., Khritankov, A.:* Improving a language model evaluator for sentence compression without reinforcement learning. In: Proceedings of the tenth international symposium on information and communication technology. pp. 92–97 (2019).
12. *Mani, I.:* Summarization evaluation: An overview. (2001).
13. *Needleman, S. B., Wunsch, C. D.:* A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of molecular biology. 48, 3, 443–453 (1970).
14. *Pires, T. et al.:* How multilingual is multilingual bert? arXiv preprint arXiv:1906.01502. (2019).
15. *Sochenkov, I. et al.:* The paraplag: Russian dataset for paraphrased plagiarism detection. In: Computational linguistics and intellectual technologies: Papers from the annual international conference “dialogue. pp. 284–297 (2017).

16. *Straka, M., Straková, J.:* Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe. In: Proceedings of the conll 2017 shared task: Multilingual parsing from raw text to universal dependencies. pp. 88–99 (2017).
17. *Straka, M., Straková, J.:* Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe. In: Proceedings of the conll 2017 shared task: Multilingual parsing from raw text to universal dependencies. pp. 88–99 Association for Computational Linguistics, Vancouver, Canada (2017).
18. *Zhao, Y. et al.:* A language model based evaluator for sentence compression. In: Proceedings of the 56th annual meeting of the association for computational linguistics (volume 2: Short papers). pp. 170–175 (2018).