Newspaper articles summarization
Nie Lubimy Poniedziałków

Michał Pastuszka
Tomasz Makowski

March 2022

1 Introduction

The summarized news article are helpful for people, who do not want to read whole article or do not have time for that. Sometimes news articles are emotional and have many subjective sentences written by the author. Those sentences should not be contained in summary, when somebody wants to know what happened in the world. Therefore, the interesting idea of creating summary is first to extract the most objective sentences, and then summary left text. In this report we will show how we can extract objective sentences, how we can summarize text and finally how this type of summarization performed on some ready datasets.

2 Theoretical introduction

2.1 Objectiveness

We decided to approach the problem of filtering objective sentences by assigning some objectivity score to each sentence. Using that, we would be able to filter only those sentences that are above a certain threshold. Distinguishing between subjective and objective texts isn't however a popular NLP task, as it's difficult to reliably label a dataset for this task. One popular source for obtaining objective sentences is to source them from Wikipedia, as encyclopedias are supposed to contain only objective information. It is more difficult, however, to obtain subjective sentences, especially if we want them to come from the same domain. One of the few ready-made methods for scoring the subjectivity of a sentence is the subjectivity score provided by the blob python package for text analysis. It is also possible to train a deep learning language model for this task. Unfortunately, probably due to a lack of a good dataset, we were unable to find a pretrained model for this task in English language (There was, however, one for Danish language). The simplest suggested approach to subjectivity detection we found was to mark sentences with words that are known to indicate subjectivity, for example words with strong emotional load, i.e. "horrible". Therefore, we decided it might be wise to use sentiment analysis methods in this task.

2.2 Summarization

Extractive methods are methods that extract key sentences from the text, without generating new ones. As an example of an extractive method, we used BERT Extractive Summarizer. This method is based on creating embeddings of sentences in text using a transformer. These embeddings are then clustered and sentences closest to each cluster centroid are extracted as parts of the summary. To use this method, the number of sentences to extract needs to be provided. It can also be given as a ratio of the entire text. More advanced methods are often based on this approach, introducing additional filtering and selection steps.

The second type of method are abstractive methods. The main idea is to get the whole text and then create a summarization with the own words. It is a similar approach that humans uses to summarize text. The methods usually use transformers to create summary. For instance, we will use PEGASUS [2] transformer, which achieves lower accuracy (47.65 ROUGE1) than the best model (PRIMER [1] with 49.9 ROUGE1) on multinews dataset. Also, on the newsroom dataset it achieves 45.98 ROUGE1, and we can't find a model with bigger score. The PEGASUS [2] model is pretrained as a transformer encoder-decoder, with objective to predict masked words or whole sentences. Its implementation on huggingface contains many pretrained models, including models fine-tuned on newsroom and multi news datasets. Thus, we will perform experiments only on test set. The fine-tuning of the model will be pointless, as one epoch will take over 1500 hours on our computers. The PRIMER [1] model differs from PEGASUS mostly by introducing two types of attention mechanism. One of them is local and uses only one document. The other one is global, and it is attention over all documents. Therefore, it is good architecture for summarization of multiple documents about the same topic as in mutlinews dataset.

3 Datasets

For reference, we decided to use the Movie Review dataset. This dataset contains 5000 sentences from rotten tomatoes movie reviews and 5000 sentences from imdb movie descriptions. The reviews are supposed to be subjective, and the descriptions objective. We used this dataset to test our methods before applying them to the summarization task.

In the case of summarization, we have chosen 2 datasets: newsroom and multinews. The biggest difference between them is that newsroom contains summary of article written by editor, whereas in the mutlinews one sample contains many articles about the same event and summaries are written by the webpage newser.com. We also identified dailymail and xsum datasets, but summaries weren't suitable for our task. In dailymail summary was few highlights from the text and in xsum the summaries was only one sentence length usually with reported speech. In both used dataset (multinews and newsroom) we performed experiments on 100 samples from the test set, as one of our methods, PEGASUS [2], used the training set to fine-tune the model. We would gladly include samples from this datasets in here, but their too large to fit in the margin.

4 Evaluation

Evaluation of a summarization task is a particularly difficult problem, as there is no one true summary for a given text. We can only use a human written summary as a comparison. To evaluate the generated summaries, we decided to use the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metrics. They are a family of metrics developed to evaluate summarization and machine translation tasks. Those metrics measure the number of matching n–grams between generated and reference sequences. In our case, this is a comparison between the text generated by a summarizing method and a provided human created summary of the text. Specifically, we used ROUGE–1, ROUGE–2 and ROUGE–L. These compare unigrams, bigrams and longest common sequence respectively. For ROUGE–1 and ROUGE–2 the number of matching n–grams is divided by the total number of n–grams in reference and generated text, giving recall and precision. In case of ROUGE–L we divide the length of the longest matching sequence by the number of unigrams (words) in corresponding texts. The value of the metric is then given as the $F-1$ score from these two values, scaled to be between 0 and 100.

5 Methods

5.1 Filtering

We decided to test three methods for filtering out subjective sentences. As this task is not very popular, we found only one method specifically designed for this. That is the subjectivity score from blob package in python. Therefore, in addition to that, we decided to test some sentiment analysis methods. We expected that subjective sentences will tend to have stronger emotional sentiment. For that, we used VADER sentiment score, as well as DistilBERT transformer pretrained for sentiment analysis. In addition to that, we tried to train a simple model that would use the outputs of these methods as features for classification. Then, we removed sentences that exceeded a certain threshold score from a given method before passing the text to a summarization method. For example, a threshold of 0.001 for transformer means that we remove sentences if the sentiment prediction probability of the model exceeds 0.999, regardless if the sentiment predicted is positive or negative. The values were inverted like this because in the movie reviews dataset objective sentences have the class 1.

5.2 Summarization

After filtering each text with one method, we apply one of the two summarization methods described before (BertExt or PEGASUS). Due to long prediction time, we decided to limit ourselves to a sample of 100 texts from each of the datasets in the experiments. We tested both algorithms with different portion of data removed by choosing different filtering thresholds. To verify the validity of our approach, we also compared the results for each filtering method with results obtained when we remove the same number of sentences from a given text chosen at random.

6 Results

Firstly, we found out that Pegasus model fine-tuning is a bad idea, since epoch will take around 1500 hours. Therefore, we analyzed following results with using pretrained models.

6.1 Objectiveness EDA

To test our subjectivity scoring methods we used the movie reviews dataset. For each method we tested, how well the score separates the two classes. In case of the linear model, the data was split into training and testing set. Other methods were unsupervised, therefore they were evaluated on the whole data. The ROC curves with AUC are shown in figure 1. As we can see, The best result was achieved by the transformer. Moreover, combining the scores as an input to a linear regression model gave results worse than using the transformer score directly. The worst result was achieved by VADER. Therefore, we decided to use Blob and transformer in further tests.

6.2 BertExt

As BertExt requires the information on how many sentences to extract. We decided that it is more reasonable that instead of choosing a fixed number of sentences for each text, we chose a ratio of sentences to keep. For example, for ratio 0.2, if the whole text had 10 sentences, we tell the model to extract 2. We compared the results for different ratios on both dataset. We tested the model on whole texts as well as text filtered with blob and transformer. We chose threshold 0.4 for blob and 0.001 for transformer, as they seemed reasonable in initial tests. The results are shown in figures 2, 3. As we can see, the results differ vastly between datasets. However, in both cases all three scores seem to peak in a specific point. Therefore, in further tests we chose to use ratio 0.02 for newsroom and 0.2 for multinews.

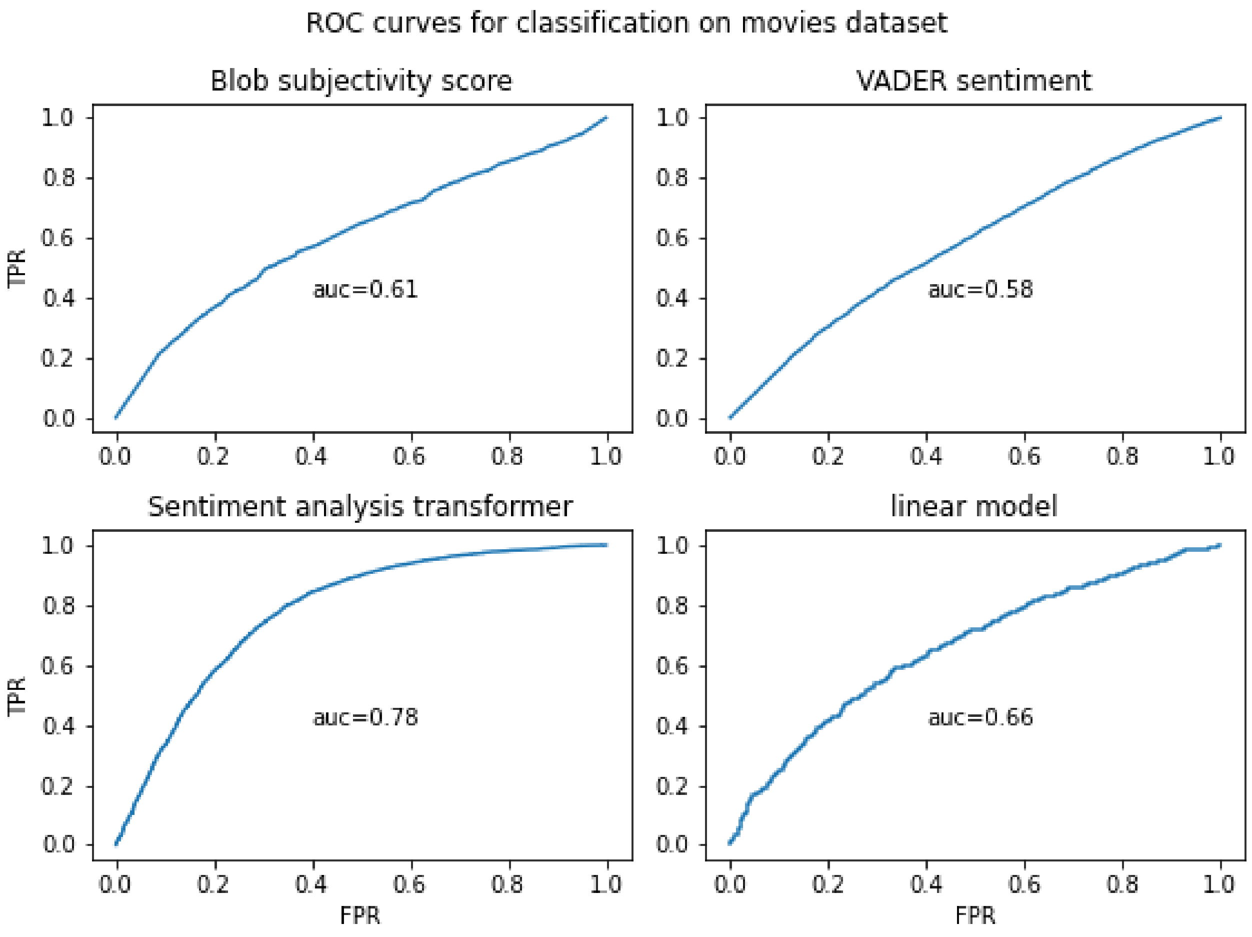

Figure 1: ROC curves on movie reviews dataset.

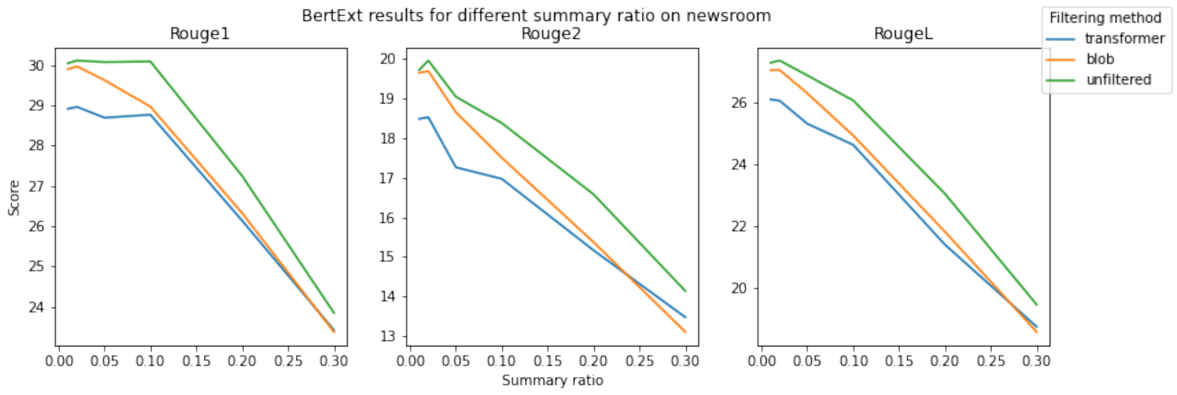

Figure 2: BertExt scores for different ratios on newsroom.

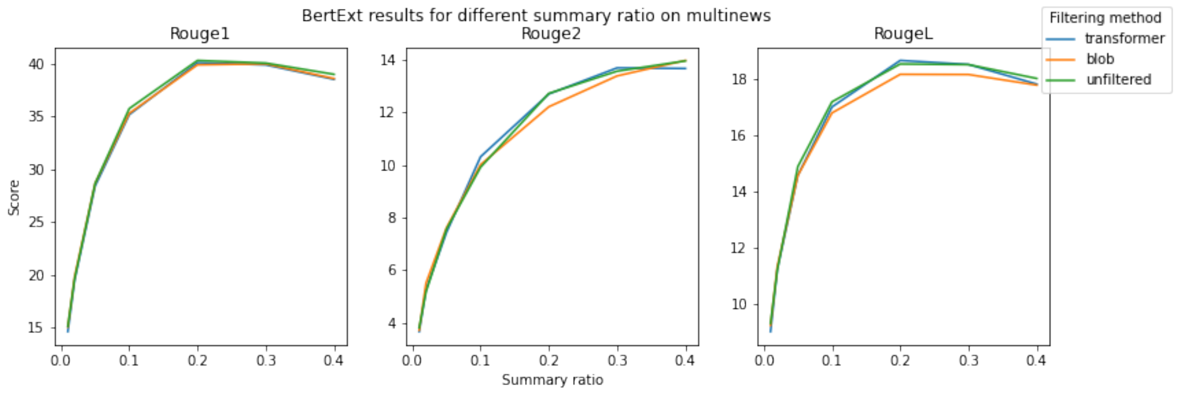

Figure 3: BertExt scores for different ratios on multinews.

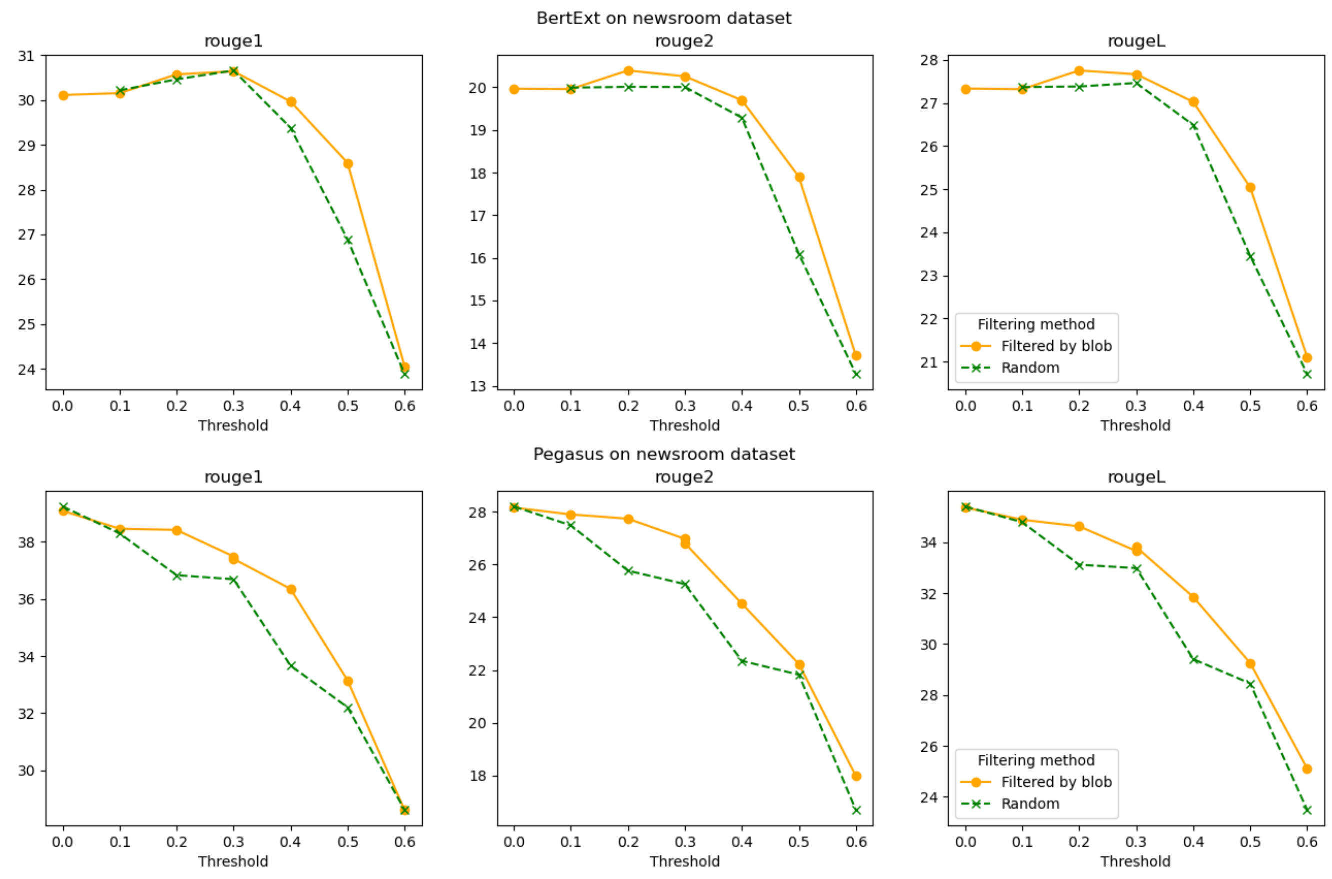

Figure 4: Blob results on newsroom dataset.

6.3 Blob

Figure 4, shows the change in accuracy of both tested models after filtering sentences with the Blob method. What's interesting is that we have managed to achieve a slight improvement in performance in case of the BertExt model. We were not able to increase the performance of Pegasus, but still the decrease in accuracy was slower than that caused by removing sentences at random. Results of the same experiment on the multinews dataset are shown in figure 5. In this case results appear to be mostly random, in particular for the BertExt model.

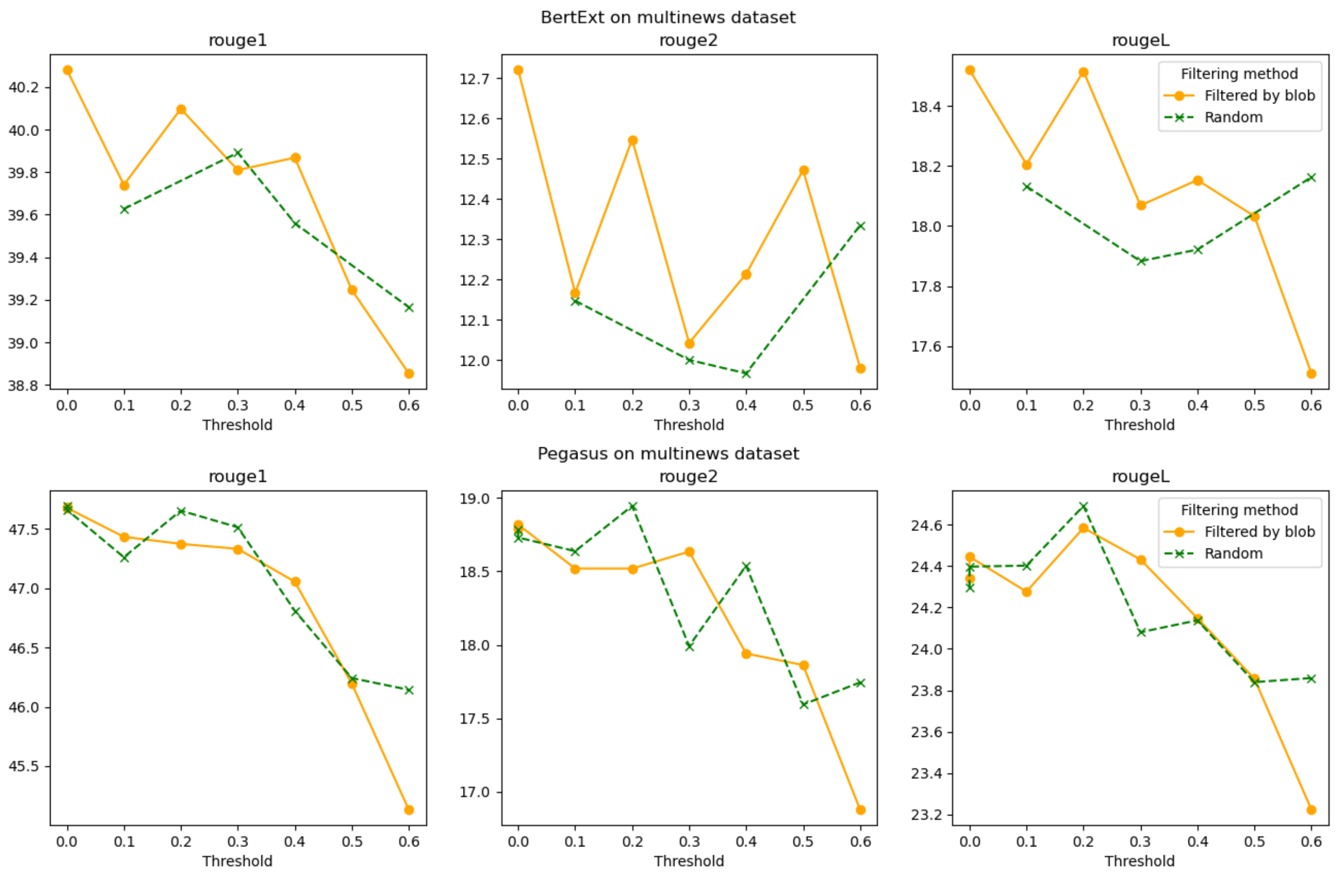

Figure 5: Blob results on multinews dataset.

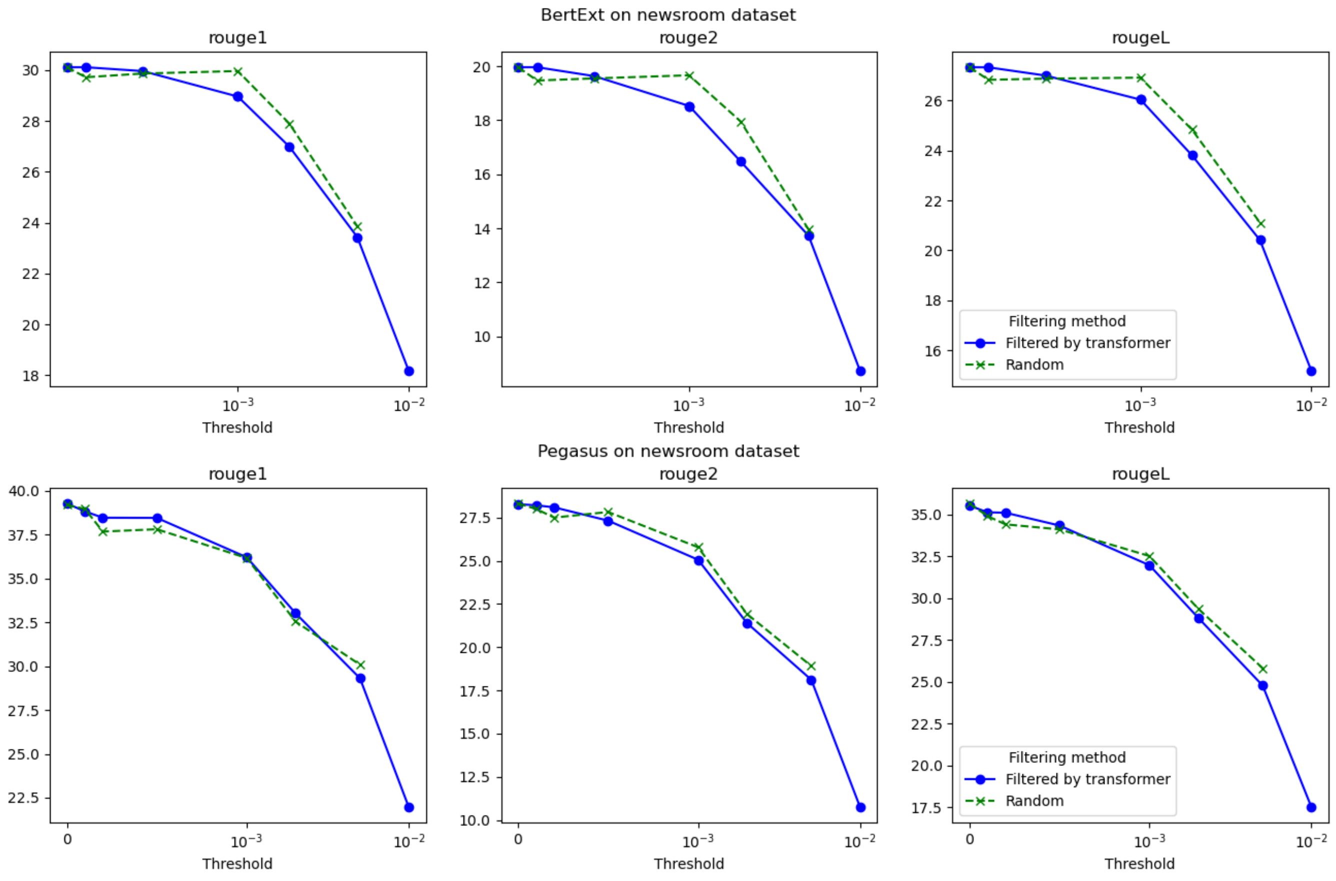

Figure 6: Transformer results on newsroom dataset.

6.4 Transformer

The experiment as described in the previous section was repeated for data filtered using the transformer. In that case, we were not able to achieve any improvement on the newsroom dataset, as shown in figure 6. Both models showed a gradual decrease in performance as more sentences were removed. In some cases, the performance was even worse than when removing sentences at random. On the second dataset 7, a small increase in performance is visible in some cases. Unfortunately, the scale of changes is very small, and comparing them with scores achieved with removing sentences at random suggests the increase is not significant. This leads to the theory, that sentences with strong sentiment might in fact contain valuable information from the article. This is probably the case with articles handling highly emotional topics, such as violent crimes.

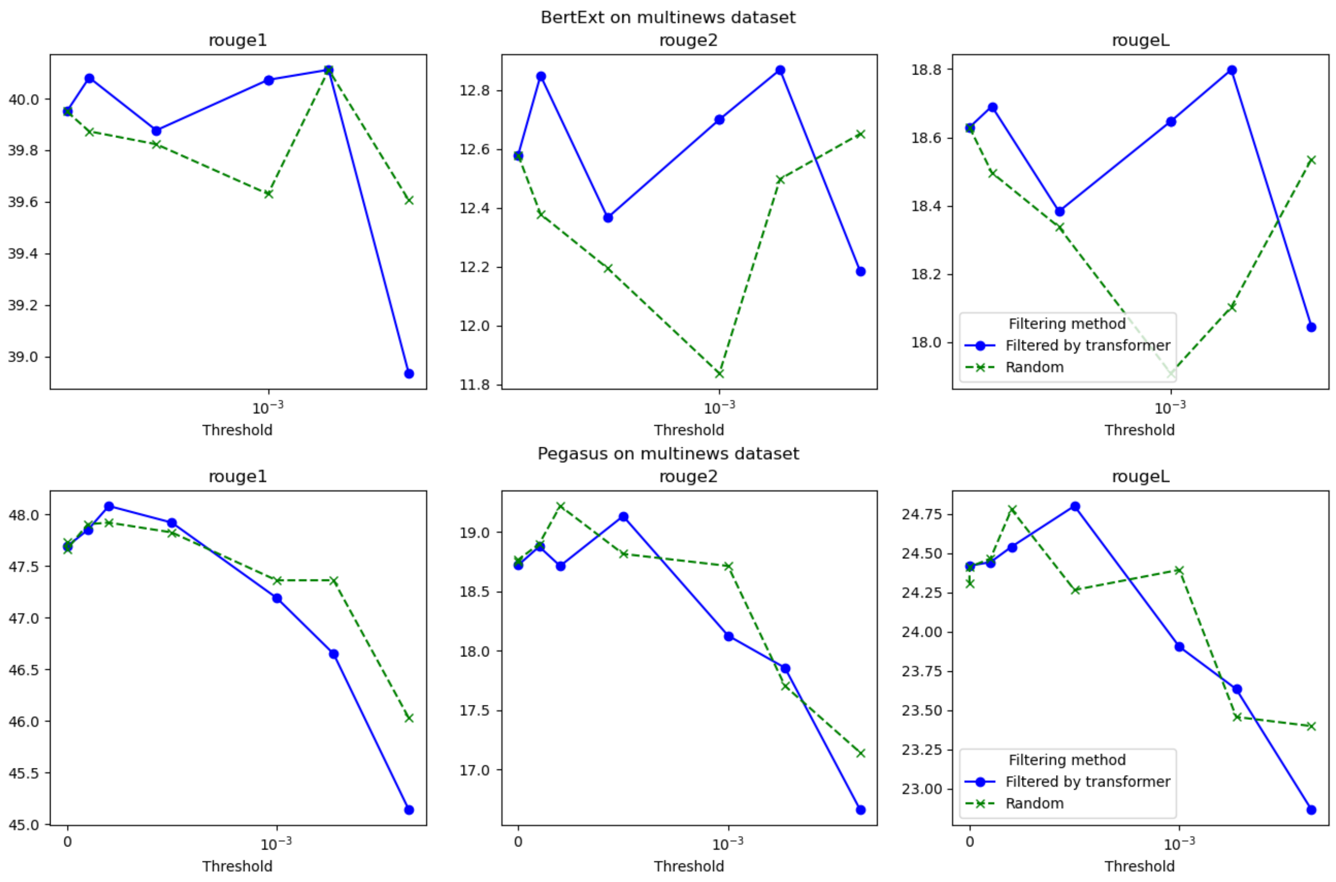

Figure 7: Transformer results on multinews dataset.

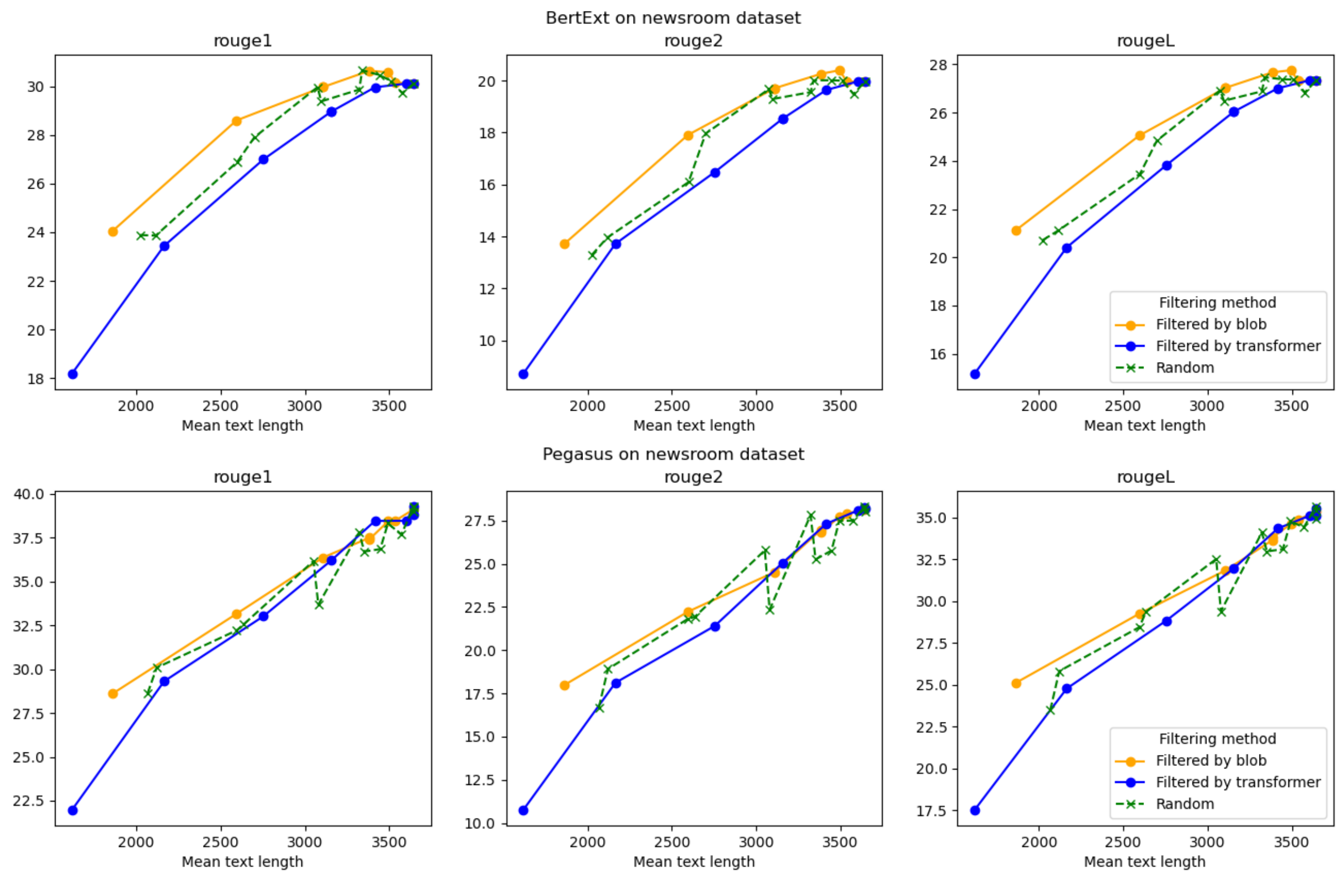

Figure 8: Results depending on filtered text length on newsroom dataset.

6.5 Number of characters

Figures 8 and 9 show how reducing the mean length of texts by various filtering methods affected the performance scores. In general, the trend shows that the performance decreases as more text is removed. There are, however, small increases as described in previous sections, particularly for Blob + BertExt on the newsroom dataset.

6.6 Heatmaps

We decided to see, how metric values change for specific texts, depending on their length before and after filtering. The results are shown in figures 10, 11 and 12. The results are from 100 multinews samples filtered by transformer and summarized by Pegasus. For each sample 6 filtering was used with values

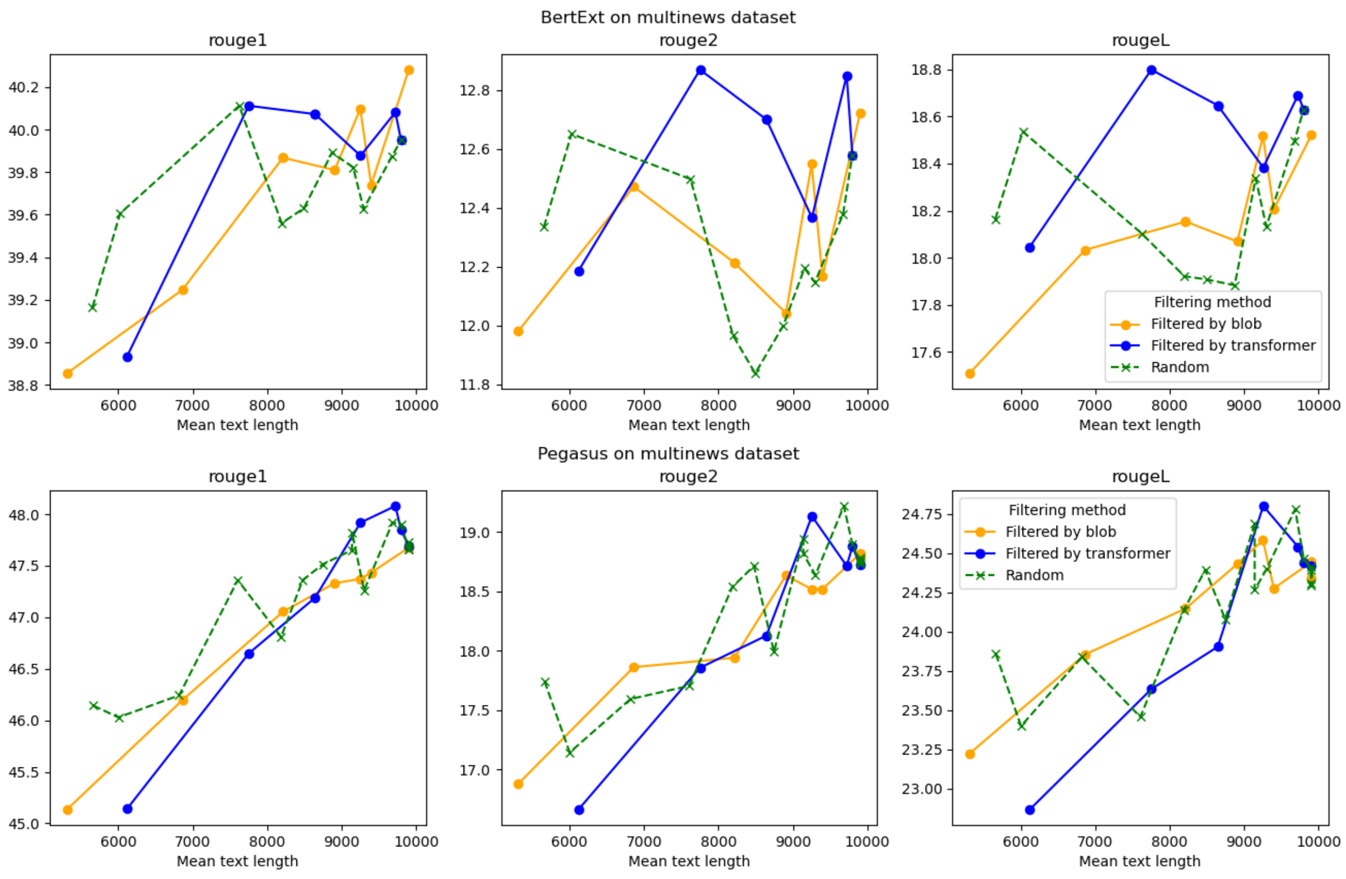

Figure 9: Results depending on filtered text length on multinews dataset.

Table 1: Final results of baseline methods and methods applied after text filtering.

dataset	method	filtering	length	rouge1	rouge2	rougeL
Multinews	BertExt	Baseline	9902.78	40.2815	12.7210	18.5211
Multinews	BertExt	Transformer 0.002	7751.77	40.1122	12.8683	18.7981
Multinews	BertExt	Blob 0.2	9251.31	40.0976	12.5480	18.5161
Multinews	Pegasus	Baseline	9902.78	47.6871	18.7212	24.4209
Multinews	Pegasus	Transformer 0.0005	9255.50	47.9188	19.1343	24.8020
Newsroom	BertExt	Baseline	3643.29	30.1140	19.9668	27.3317
Newsroom	BertExt	Blob 0.2	3493.73	30.5741	20.3963	27.7525
Newsroom	Pegasus	Baseline	3643.29	39.0886	28.1645	35.3626
Newsroom	Pegasus	Blob 0.3	3384.18	37.4897	26.9782	33.6508

0.001, 0.002, 0.005, 0.0001, 0.0002, 0.0005 so in total there are 600 observations. Firstly, one can observe that the lowest scores are both for the longest initial text and the shortest initial text. Then, we can observe that the biggest values (easiest to observe on figure 10) are for rather shorter initial text, but it is hard to find a general conclusion. Also, for many initial lengths we can see that there are not significant decrease with shorter text after filtering, but sometimes there is also increase. For ROUGE-L score (figure 12) we can see that longer initial length means lower accuracy, since it is easy in this measure to split to longer proper sequences when the text is longer.

6.7 General results

Our best results for different extraction and filtering methods are contained in the table 1. As mentioned before, in some cases we were able to achieve a slight improvement over the baseline. Additionally, we can see that for transformer filtering, combined with BertExt on multinews dataset, the portion of the text removed was quite large. Despite that, ROUGE1 was reduced only slightly, while the other scores showed a small rise. This serves as a good proof of concept for this approach.

7 Summary

Overall, the proposed concept seems interesting, but requires further research. Some experiments showed an improvement in achieved ROUGE scores after removing some sentences. This shows promise in the approach, but one need to consider that tests were performed with the usage of only 100 dataset samples. The main hurdle was the limited access to methods of detecting subjective sentences. If we were able to, for example, train a deep learning model for this specific task, perhaps the results would show a clearer improvement over the

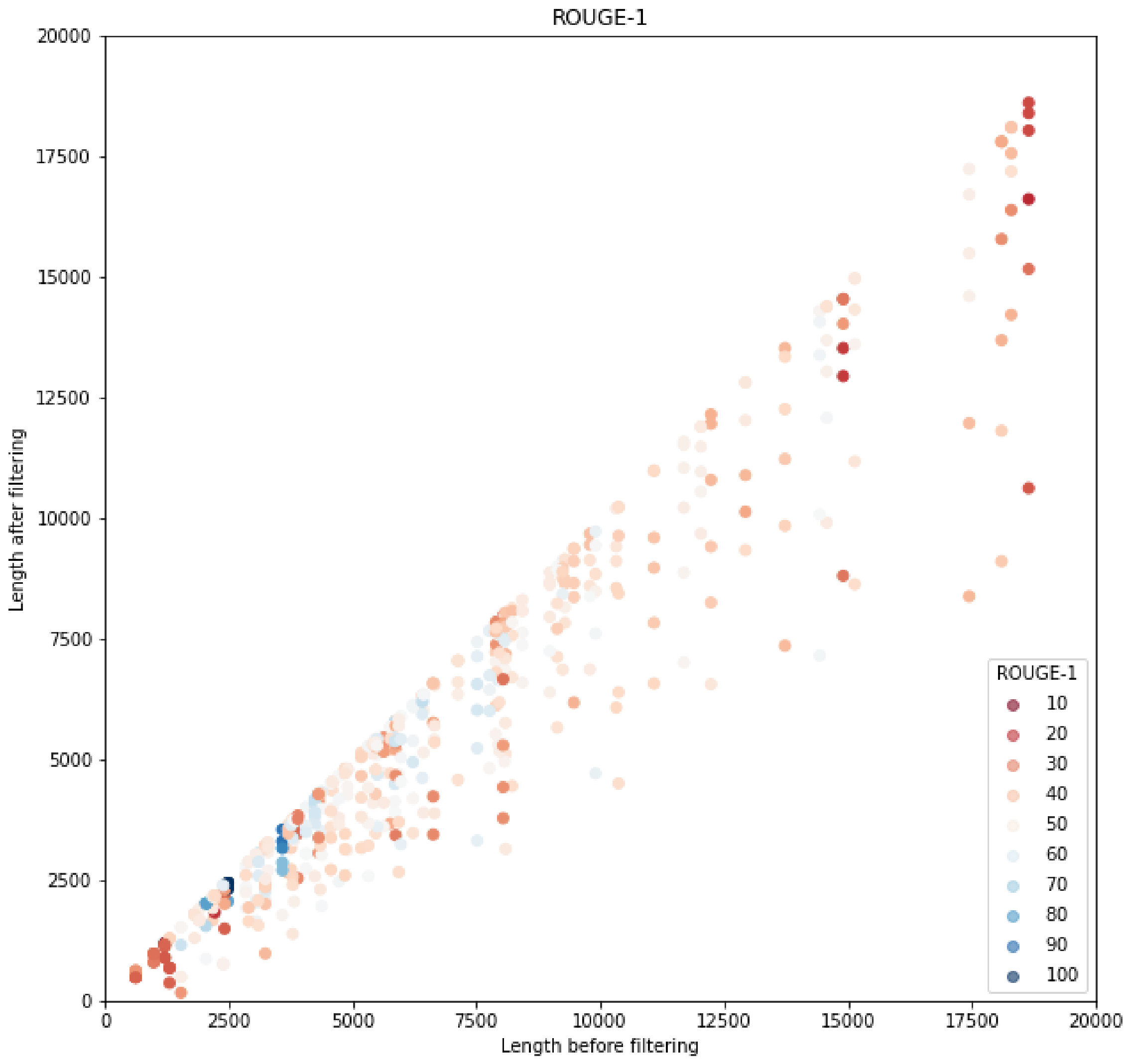

Figure 10: ROUGE–1 depending on full text length and length after filtering.

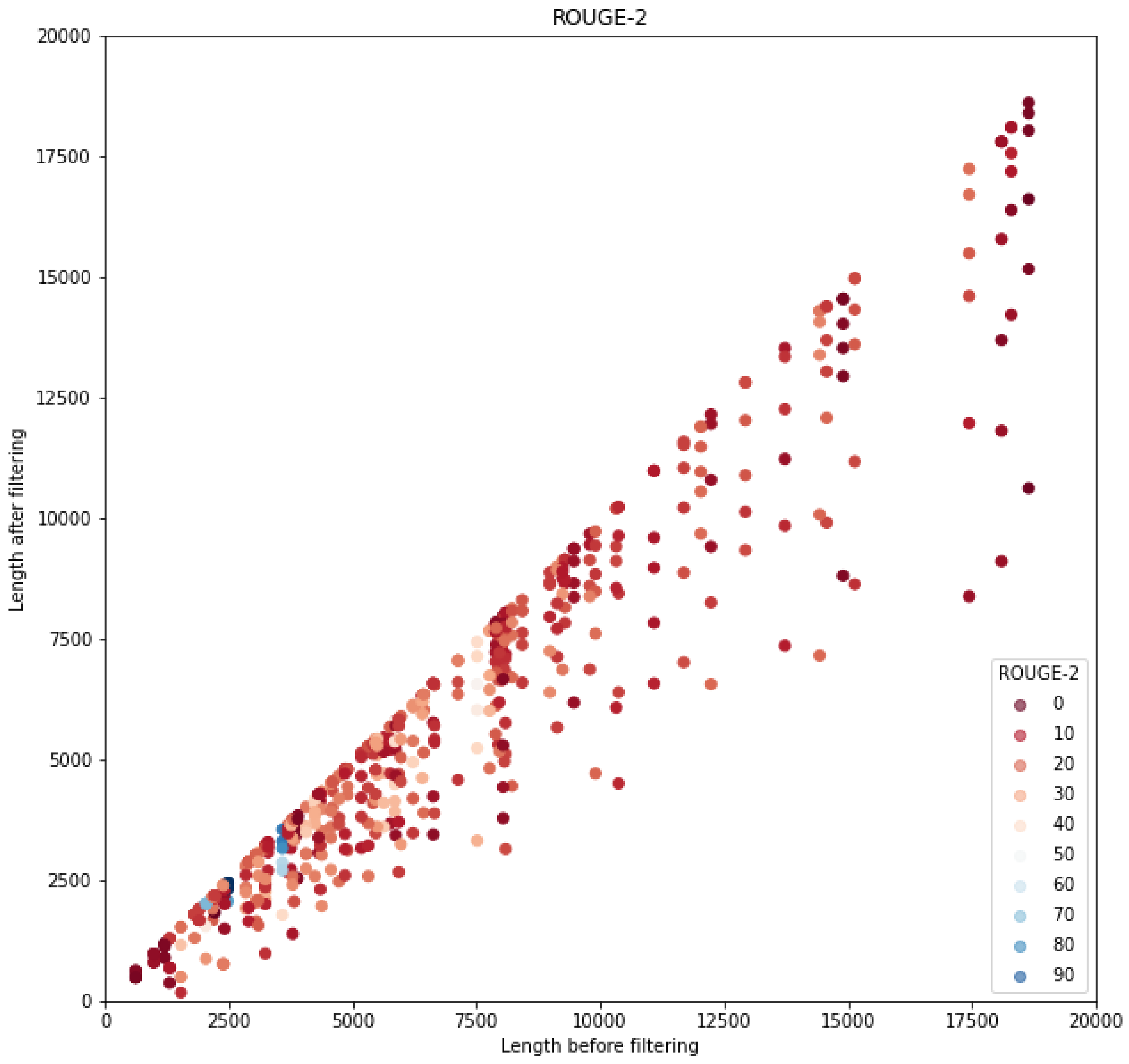

Figure 11: ROUGE–2 depending on full text length and length after filtering.

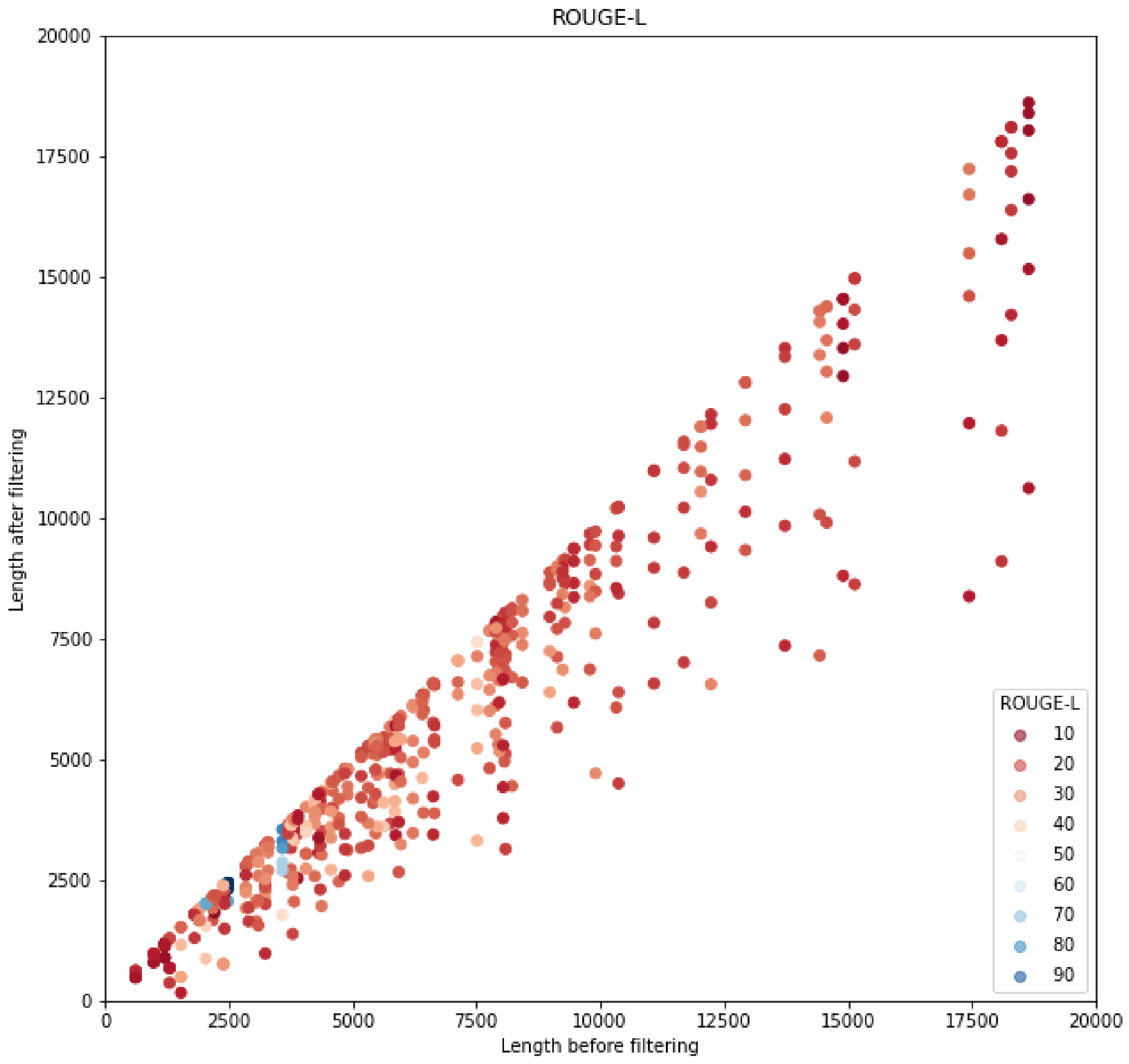

Figure 12: ROUGE–L depending on full text length and length after filtering.

baseline. This would firstly require preparing a good dataset from the news domain. Secondly, deep learning NLP models take a lot of computing power to train. The other possibility of sentence filtering is to check if the sentence is emotional or not and filter less emotional sentences. Although, it also requires dataset creation.

The length of summaries analysis showed that longer texts are harder to summarize, since results were lower. Nevertheless, too short text is also harder to summarize.

References

[1] W. Xiao, I. Beltagy, G. Carenini, and A. Cohan, "Primer: Pyramid-based masked sentence pre-training for multi-document summarization," *arXiv preprint arXiv:2110.08499*, 2021.

[2] J. Zhang, Y. Zhao, M. Saleh, and P. Liu, "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization," in *International Conference on Machine Learning*, PMLR, 2020, pp. 11 328–11 339.