Team 26: Text summarization

April 25, 2021

Abstract

In our project, we have decided to create text summarizer with BERT. There is already a Bert Extractive Summarizer -package. It's based on following article: https://arxiv.org/ftp/arxiv/papers/1906/1906.04165.pdf This package is intended for lecture-summarization, and our goal is to extend and fine-tune this model for news or scientific articles. Even though foundational work is already done for the package, there's still much to customize such as tokenizer and model. Our target language is english. Summarization tools such as Bilingual Evaluation Understudy (BLEU) and Recall-Oriented Understudy for Gisting Evaluation (ROUGE) are available, and part of the project is to try to evaluate the texts automatically and manually.

1 Introduction

Automatic summarization is the process of shortening a set of data computationally, to create a subset (a summary) that represents the most important or relevant information within the original content. There are two main approaches to automatic summarization (independently of the application domain, e.g. text, images, video etc.):

- Extraction-based or extractive summarization
- Abstraction-based or abstractive summarization

When it comes to text documents, summarization is closely related to data compression and information understanding. The ability to produce coherent, well-structured summaries has the potential to transform efficiently the way that discovery systems work, as well as help human readers in skimming large datasets of text documents. That is why automatic summarization is considered one of the most important, yet least solved, tasks in NLP and a method that will transform the way people consume information on the Internet. In conclusion, applying text summarization reduces reading time, accelerates information retrieval and increases the amount of useful, dense information. In our case, we will deal with extractive summarization where a system produces summaries by choosing a subset of the initial text.

2 Background

The first summarization techniques go back already more than 50 years to Luhn's and Edmundson's seminal papers on automatic summarization (1958 and 1969 respectively, [4], [1]). Early work in the field dealt with single document summarization (news story, scientific articles etc.) Later, multi document summarization was applied in big data clusters to provide a coherent and brief digest to the users.

3 Methods

4 Experiments

5 Results

Evaluation was done on newsroom dataset [2]. The model inference however is extremely slow, and only a subset of samples was chosen to be evaluated. The sample size was chosen to be 500 at randomly, which resulted in tolerable running times (¡ 60 minutes).

5.1 General text properties

Two different BERT models were tested, BERT$_{LARGE}$ and BERT$_{BASE}$. These models were compared against baseline model Lede-3 and GPT-2. GPT-2 was chosen to be compared against BERT models, because in the original study [5] they noticed that BERT should be generating more representative embeddings of the sentences. Also GPT-2$_{LARGE}$ was chosen as one model to be evaluated. GPT-2$_{LARGE}$ has 774 million parameters, which is roughly the double of parameters in BERT$_{LARGE}$ and GPT-2$_{MEDIUM}$. Also GPT-2-XL and new GPT-NEO (GPT-3 replication) would've been interesting comparisons, but unfortunately they were too large to fit on GPU.

The python package bert-extractive-summarizer [5] uses only sentences from original text, and the amount of sentences is defined by either fixed ratio, or amount of sentences. In this case the ratio was defaulted to 0.2, which means that we use 20% of the sentences from original text. We can calculate empirical distribution of the ratios seen in reference summaries as displayed in Figure 1.

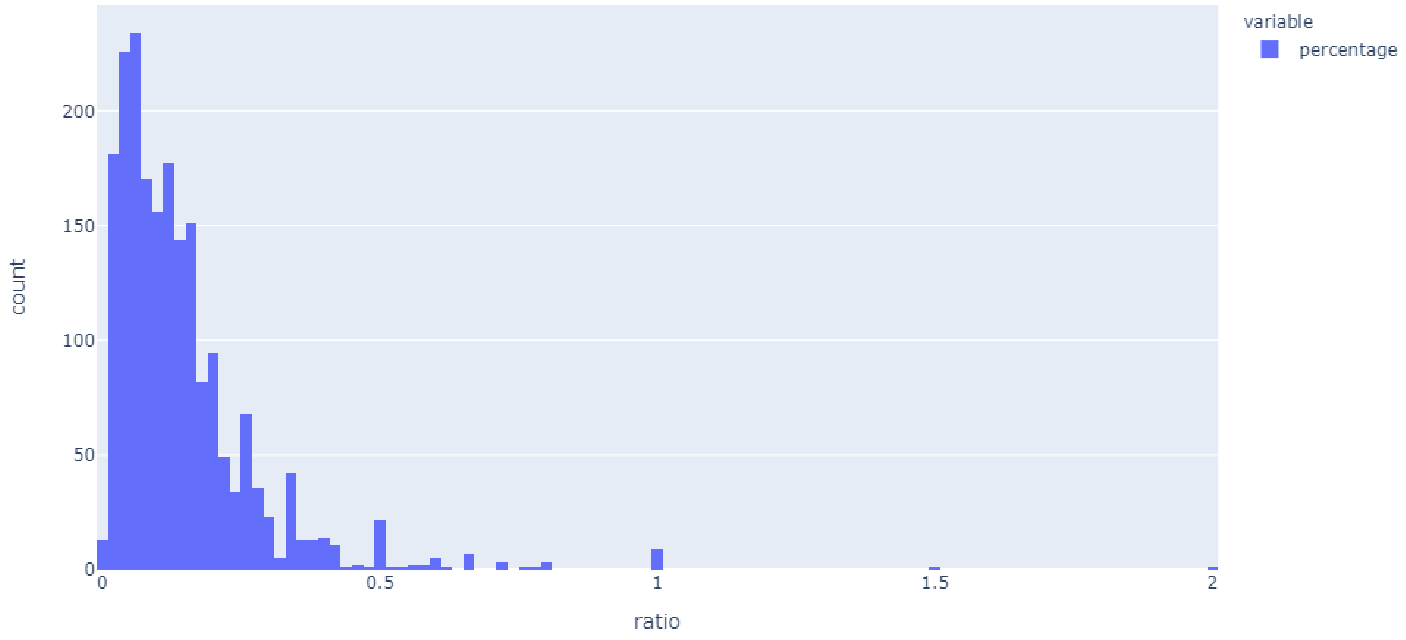


Figure 1: Distribution of observed ratios in the data

Here we can see that the mean is around 0.15. We can also plot what length of sentences there are as in following Figure 2.

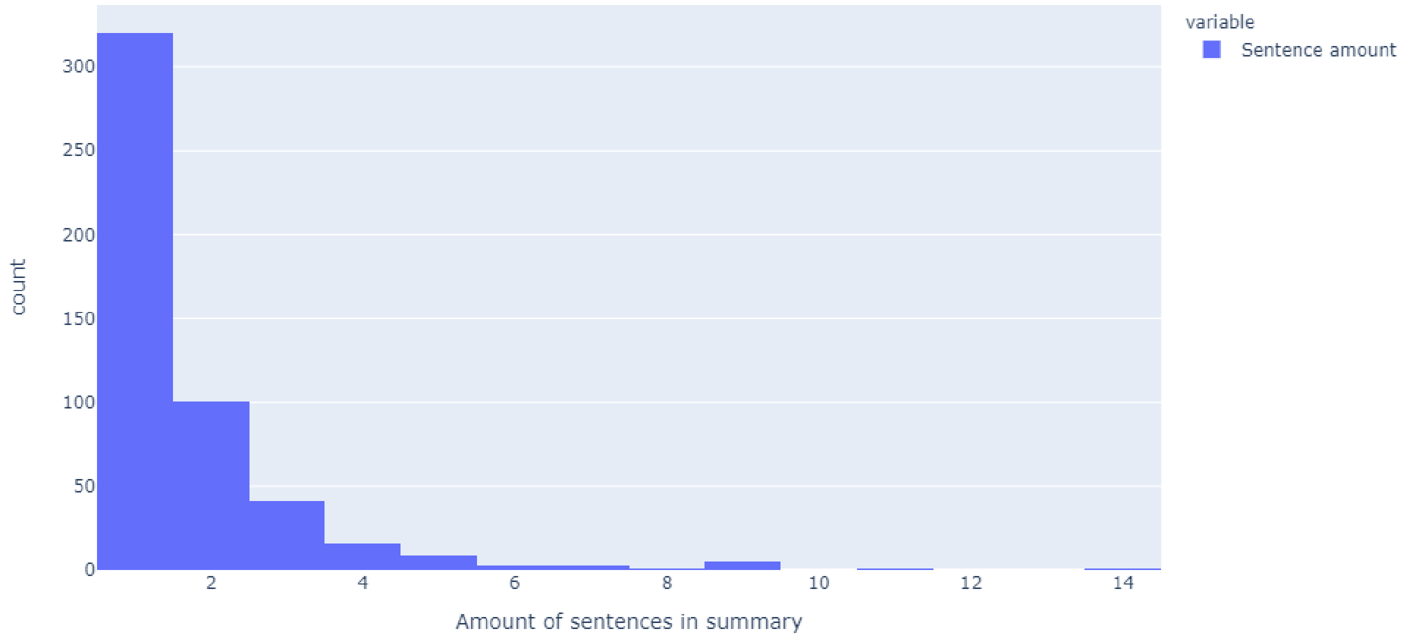
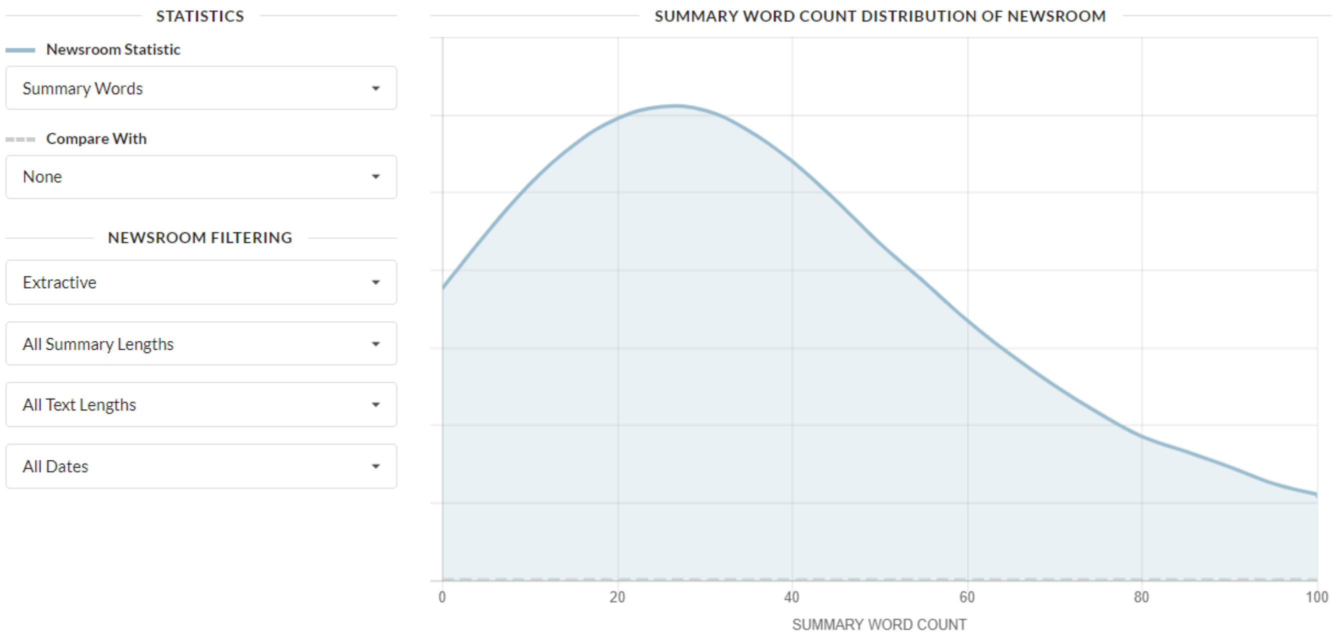


Figure 2: Distribution of observed ratios in the data

Here we actually see that there are multiple summaries which have sentence amount of 1. This is important to remember because it might favor Lede-3 classifier due to shortness of text, as we see later.

We can also view the actual summary distribution statistics from the dataset website summari.es [2]. By using their visualization tool, we get following distribution.


STATISTICS

Newsroom Statistic
Summary Words

Compare With
None

NEWSROOM FILTERING

Extractive

All Summary Lengths

All Text Lengths

All Dates

Figure 3: Distribution of words in summaries

If we plot amount of words seen in our samples, we get similar looking plot as in following Figure.

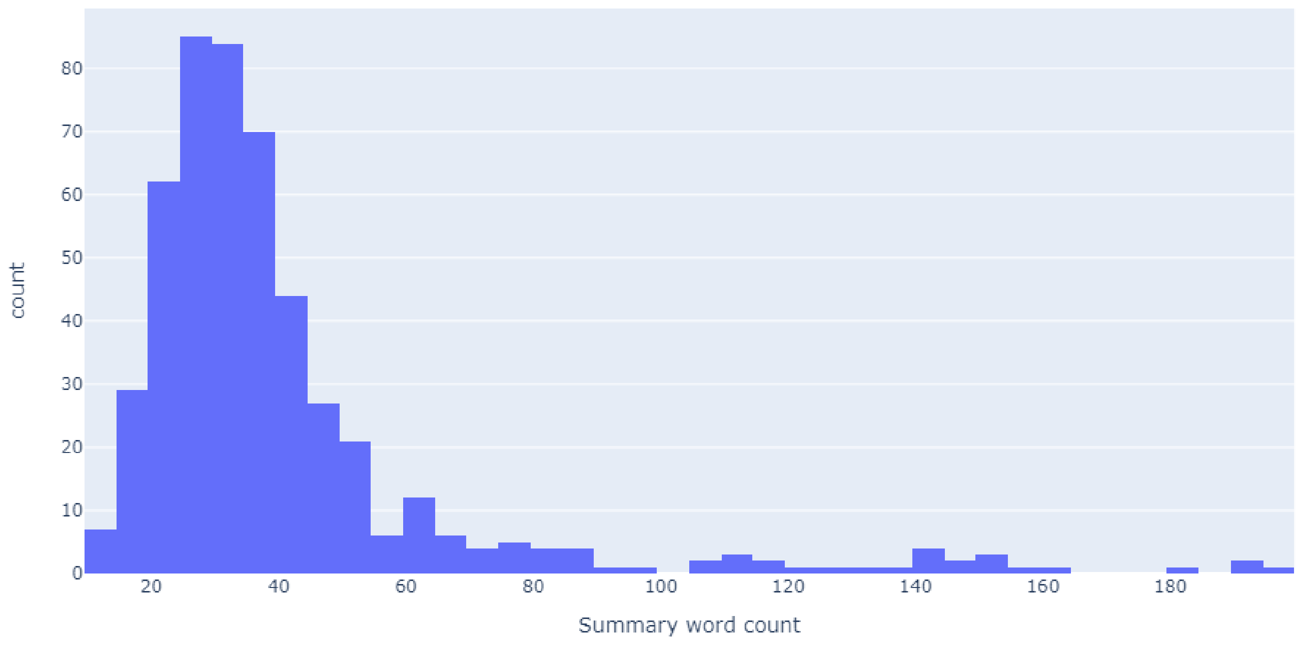


Figure 4: Distribution of words in summaries

From this it seems that newsarticle summaries tend to be short.

5.2 Recall-Oriented Understudy for Gisting Evaluation (Rouge)

ROUGE is a metric which determines the quality of summary. In this experiment we used ROUGE-N and ROUGE-L metrics. ROUGE-N metric measures n-gram overlapping between generated and reference summaries. ROUGE-L metric compares the longest common subsequence between generated and reference summaries. ROUGE recall can be defined as following [3]:

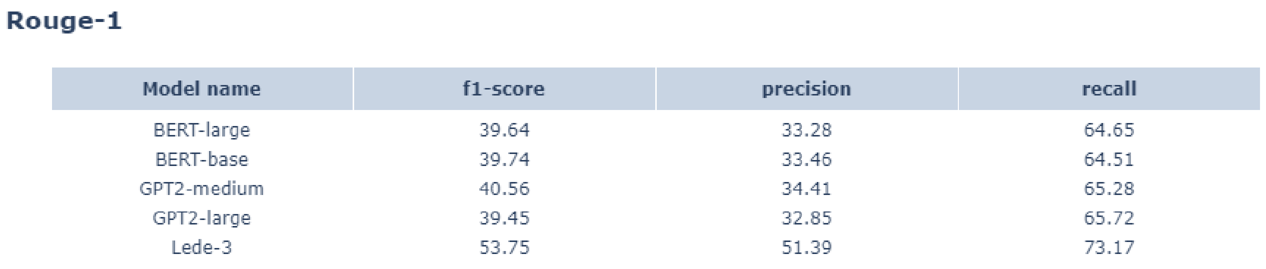
$$\text{ROUGE-recall} = \frac{\text{n-overlapping-words}}{\text{n-words-in-reference-summary}}$$

and ROUGE precision as following:

$$\text{ROUGE-precision} = \frac{\text{n-overlapping-words}}{\text{n-words-in-generated-summary}}$$

ROUGE-f1 score used in the coding implementation is similar to normal f1-scores, this time just used with ROUGE-precision and ROUGE-recall. [3]

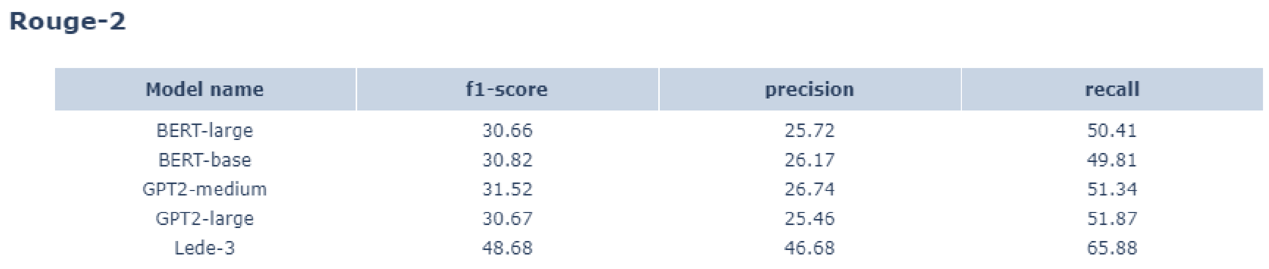
Rouge scores seem to be dominated by Lede-3. Lede-3 is known to have strong performance, comparable with state-of-art methods as discussed in [2]. Otherwise the models are performing similarly, although GPT-2_{MEDIUM} is achieving slightly higher scores. All of the rouge scores are seen in Figures 5, 6 and 7 below.



Rouge-1

Model name	f1-score	precision	recall
BERT-large	39.64	33.28	64.65
BERT-base	39.74	33.46	64.51
GPT2-medium	40.56	34.41	65.28
GPT2-large	39.45	32.85	65.72
Lede-3	53.75	51.39	73.17

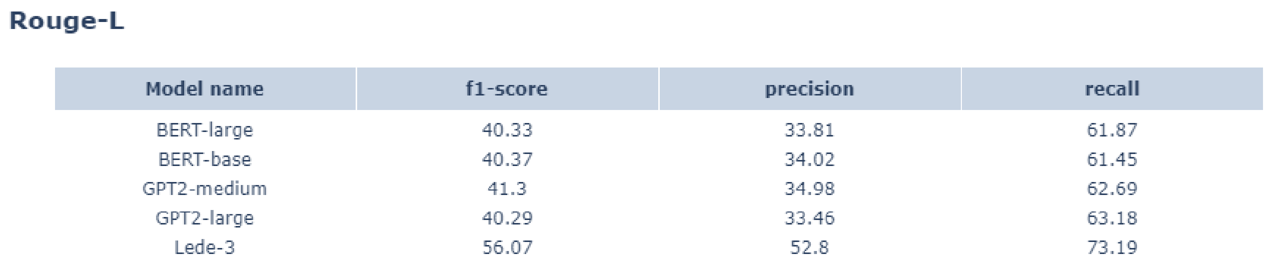
Figure 5: Rouge-1 scores with ratio=0.2



Rouge-2

Model name	f1-score	precision	recall
BERT-large	30.66	25.72	50.41
BERT-base	30.82	26.17	49.81
GPT2-medium	31.52	26.74	51.34
GPT2-large	30.67	25.46	51.87
Lede-3	48.68	46.68	65.88

Figure 6: Rouge-2 scores with ratio=0.2



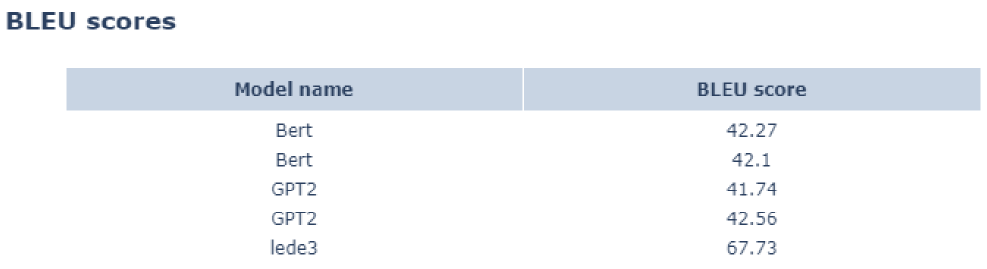
Rouge-L

Model name	f1-score	precision	recall
BERT-large	40.33	33.81	61.87
BERT-base	40.37	34.02	61.45
GPT2-medium	41.3	34.98	62.69
GPT2-large	40.29	33.46	63.18
Lede-3	56.07	52.8	73.19

Figure 7: Rouge-L scores with ratio=0.2

5.3 Bilingual Evaluation Understudy (BLEU)

We can also take a look at BLEU scores for each model. BLEU measures how much the n-grams in our generated summary appears on the reference summaries. BLEU performs similarly to ROUGE-precision, except BLEU introduces brevity penalty, which penalizes generated summaries that are shorter than reference summaries. As Lede-3 summaries are always of length 3, we can see how much Lede-3 is actually punished by looking at how many summaries have 3 sentences or more from the summary distribution in Figure 2. We see that majority of the sentences are under 3. This means that Lede-3 sentences are not punished, and it actually provides really good performance on this dataset.

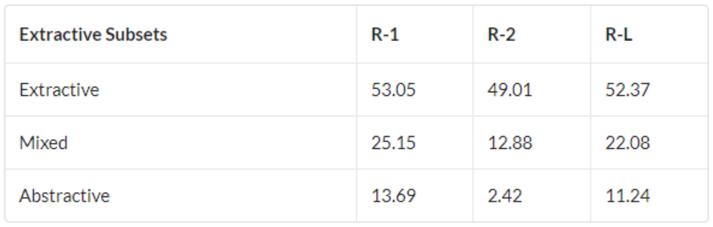

BLEU scores

Model name	BLEU score
Bert	42.27
Bert	42.1
GPT2	41.74
GPT2	42.56
lede3	67.73

Figure 8: BLEU scores with 4-grams

5.4 Comparison against other algorithms

So far we have compared our models to each other, but to understand how well this compares to other models we can look at summari.es 'Evaluate' tool. There is provided nice interactive tool to get different algorithms scores against each other. We can use F-score for comparison because it combines both recall and precision. First let's see the performance of Lede-3 to see how well our subset of data represents the whole data. In Figure 9 we see the Lede-3 performance on the whole extractive data.

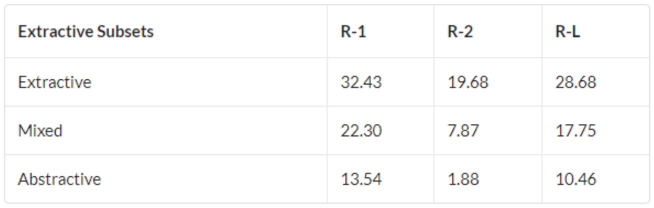


Extractive Subsets	R-1	R-2	R-L
Extractive	53.05	49.01	52.37
Mixed	25.15	12.88	22.08
Abstractive	13.69	2.42	11.24

Figure 9: Lede-3 on full data

Comparing to our Lede-3 F1-scores in Figures 5, 6 and 7 we see that the performance is almost identical with error margin of 1 unit. This gives us confidence in our choice of subset of data, and it makes comparing model performance more reliable.

Next we can compare our models to TextRank performance, which is known to be one of the best-performing models. In following Figure 10 we see that our models actually outperform TextRank in every ROUGE score in F1-metrics by a large margin, up to 10 units. This is a good indicator that our models are indeed very good, but Lede-3 is simply great choice for news articles.



Extractive Subsets	R-1	R-2	R-L
Extractive	32.43	19.68	28.68
Mixed	22.30	7.87	17.75
Abstractive	13.54	1.88	10.46

Figure 10: TextRank performance on full data

All in all, we can say that our models performed well on the data considering that they are producing word embeddings without any fine-tuning as general purpose models. Our performance would most likely ramp up if we decided to fine-tune our models to summarization tasks.

6 Discussion

In this experiment we saw how well extractive text summarization works if we choose sentences based on their expressiveness in embedding space. We used K-means clustering to cluster embeddings that are encoded from the model. We used Lede-3 as a baseline, and also compared GPT-models to see how well they produce embeddings compared to BERT. We found out that Lede-3 is the most accurate summarization technique for our data. Although Lede-3 is very simple algorithm and it has no understanding of the underlying text, it seems like the texts are written in such way that first sentences produce most important aspects of the text. This is interesting realization, and in the future it would be interesting to try to fine tune some deep learning model to see, whether it actually is able to outperform Lede-3 or not.

In this experiment it was also interesting to see how four state-of-art models performed when extracting embeddings without fine tuning. Most interesting note was that scaling the model (GPT-2_{LARGE}) didn't necessarily result in better embeddings with respect to our task. The models were also compared to well known TextRank, which they outperformed by large margin in F1-scores. This gives confidence in large general purpose models, as they were not fine-tuned for this task.

The clustering approach was used when summarizing lecture videos in [5], where this approach is more viable because lectures aren't as well structured as news articles. Although the algorithm was proposed when thinking of this type of lecture summarization, it performed surprisingly well also on news articles.

7 Division of labor

8 Appendix

References

[1] H. P. Edmundson. New methods in automatic extracting. *J. ACM*, 16(2): 264–285, Apr. 1969. ISSN 0004-5411. doi: 10.1145/321510.321519. URL `https://doi.org/10.1145/321510.321519`.

[2] M. Grusky, M. Naaman, and Y. Artzi. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 708–719, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. URL `http://aclweb.org/anthology/N18-1065`.

[3] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[4] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2), 1958. URL `http://www.research.ibm.com/journal/rd/022/luhn.pdf`.

[5] D. Miller. Leveraging BERT for extractive text summarization on lectures. *CoRR*, abs/1906.04165, 2019. URL `http://arxiv.org/abs/1906.04165`.