Text summarization presentation

Hilei Pavlo

Text summarization topic overview

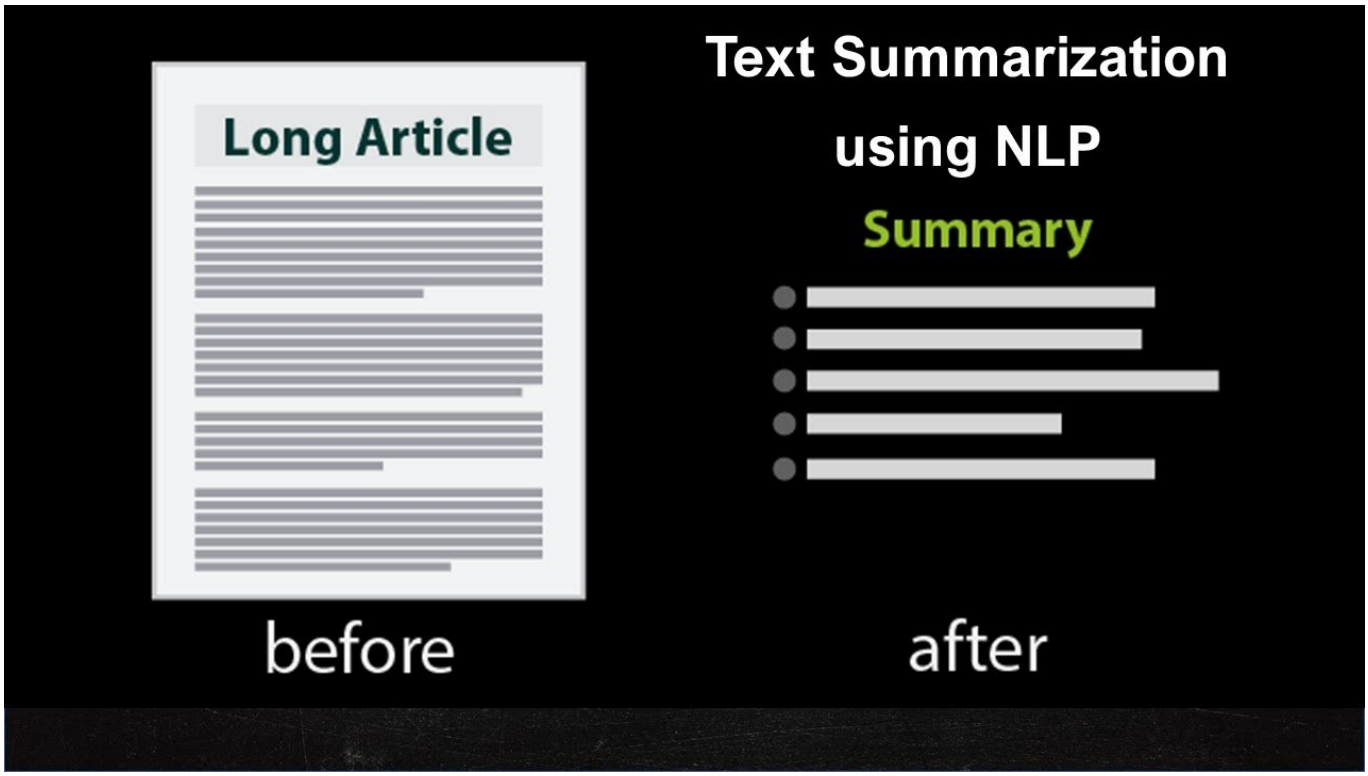
Text summarization - the task of producing a shorter version of a document while preserving its important information.[1]

Extractive approach

Rely on extracting several parts, such as phrases and sentences, from a piece of text and stack them together to create a summary

Abstractive approach

Use advanced NLP techniques to generate an entirely new summary. Some parts of this summary may not even appear in the original text.



Metrics

BLEU

$$\text{n-gram precision} = p_n = \frac{\text{number of n-gram matches}}{\text{number of n-grams in generation}}$$

$$\text{BLEU-N} \sim \sqrt[N]{p_1 \cdot p_2 \cdots p_N}$$

ROUGE

$$\text{ROUGE-N recall/precision} = r_n = \frac{\text{number of n-gram matches}}{\text{number of n-grams in reference/generation}}$$

$$\text{ROUGE-N fmeasure} = \frac{p_N \cdot r_N}{p_N + r_N}$$

$$\text{ROUGE-L recall/precision} = \frac{LCS(gen, ref)}{\text{number of words in reference/generation}}$$

LCS - longest common sequence

Datasets

- CNN/Daily mail 270K doc-sum pairs
  - len(summary) 3-4 sentences
- XSum 200K doc-sum pairs
  - len(summary) == 1
  - BBC articles

Classical, extractive approaches

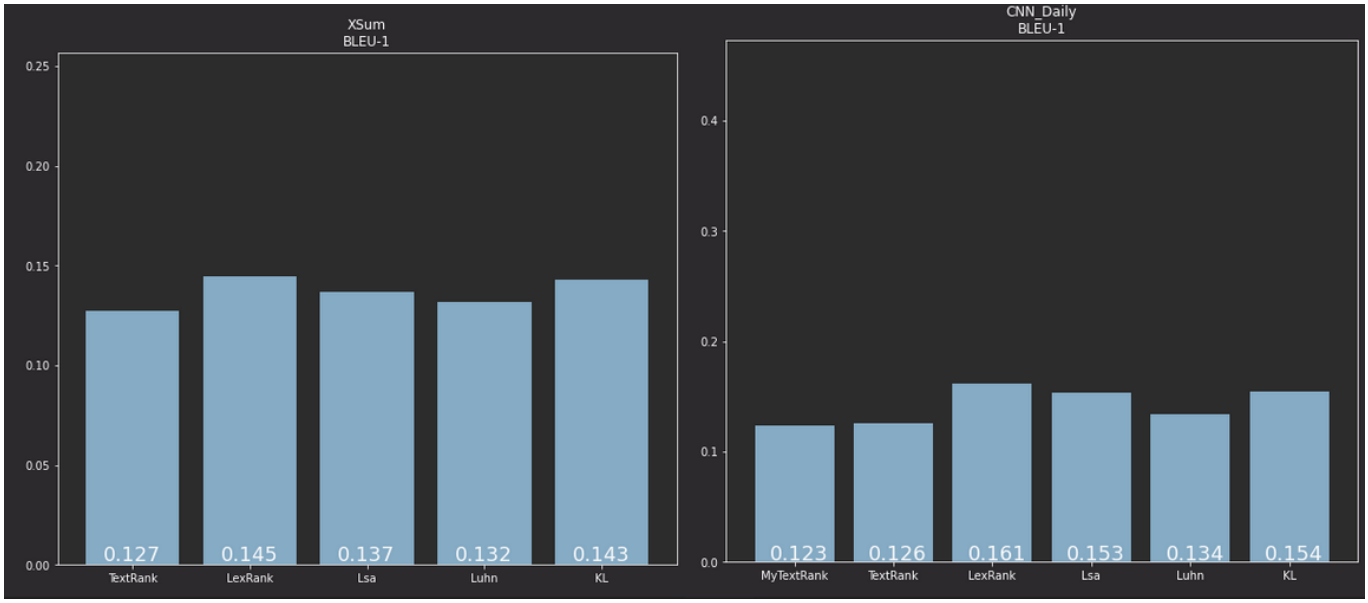
- TextRank
- Luhn
- LSA (Latent semantic analysis)
- LexRank
- KL-sum
- MyTextRank (GloVe embedings + pagerank algorithm)

GloVe - Global Vectors for Word Representation

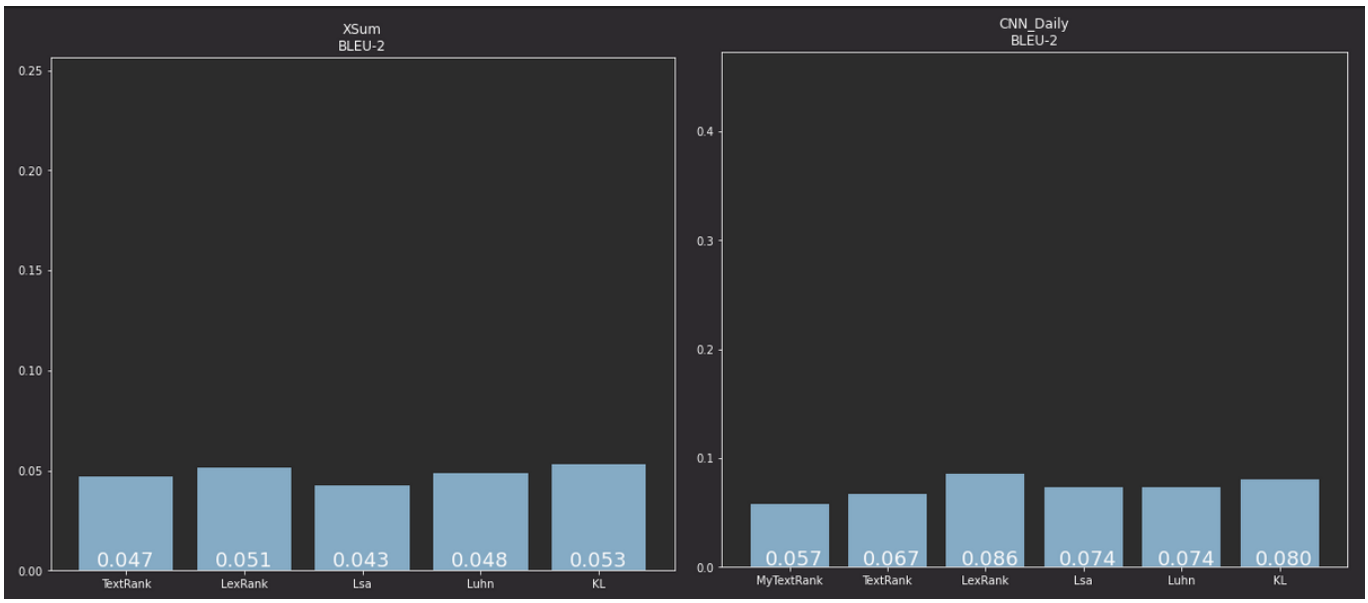
Preprocessing

```
# 3. preprocessors
preprocessor_strong = CorpusPreprocessor(name="strong").\
    add(to_lower()).\
    add(expand_contractions()).\
    add(replace_by_regex([(EMAIL_REGEX, "EMAIL")])).\
    add(remove_by_regex([r"\d"])).\
    add(remove_symbols(string.punctuation.replace('.', ''))).\
    add(lemmatize()).\  # car, cars, car's, cars' -> car
    add(remove_words(set(stopwords.words('english'))))

preprocessor_mid = CorpusPreprocessor(name="mid").\
    add(to_lower()).\
    add(expand_contractions()).\  # don't => do not
    add(replace_by_regex([(EMAIL_REGEX, "EMAIL")])).\
    add(remove_by_regex([r"\d"])).\  # remove numbers
    add(remove_symbols(string.punctuation.replace('.', '')))

preprocessor_simple = CorpusPreprocessor(name="simple").\
    add(to_lower()).\
    add(replace_by_regex([(EMAIL_REGEX, "EMAIL")])).\
    add(remove_symbols(string.punctuation.replace('.', '')))

preprocessor_no = CorpusPreprocessor(name="nothing")
```

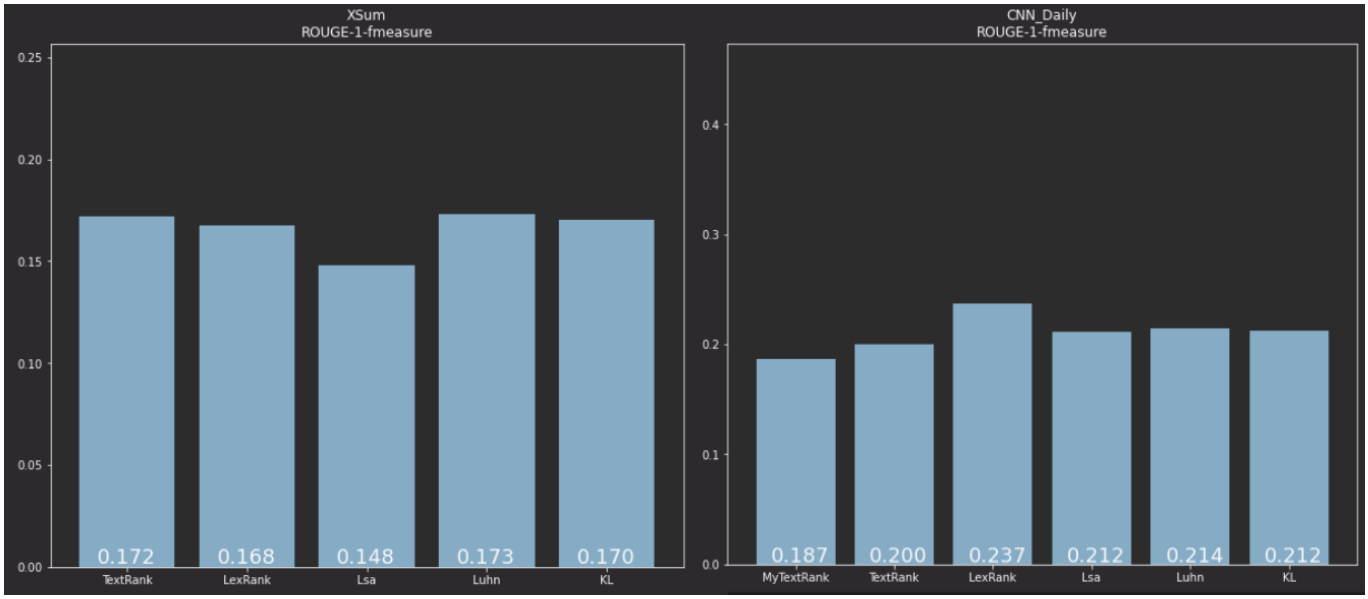
Metrics

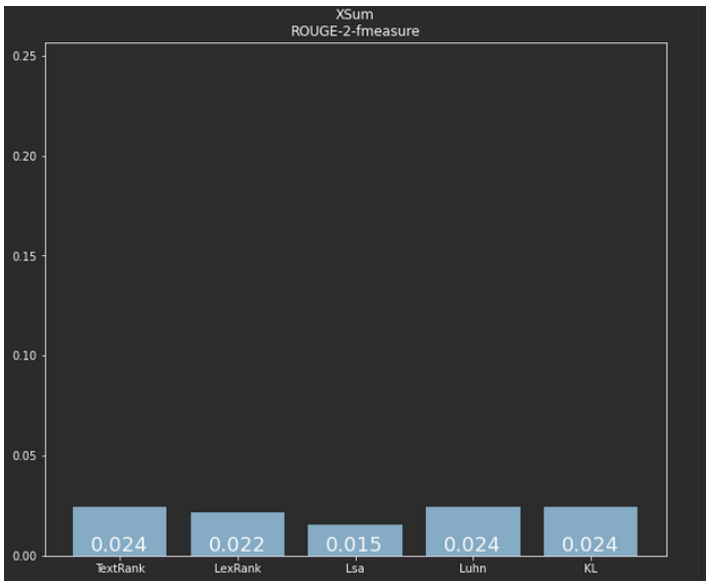

XSum
BLEU-1
0.25
0.20
0.15
0.10
0.05
0.00
0.127
0.145
0.137
0.132
0.143
TextRank
LexRank
Lsa
Luhn
KL

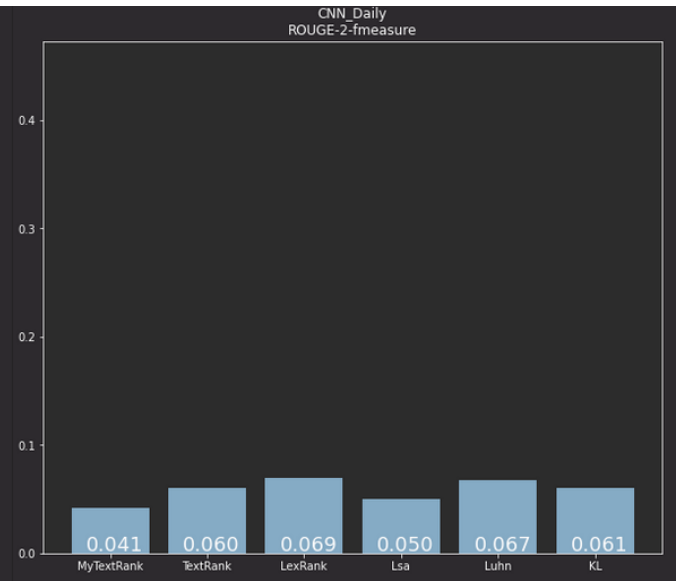
CNN_Daily
BLEU-1
0.4
0.3
0.2
0.1
0.0
0.123
0.126
0.161
0.153
0.134
0.154
MyTextRank
TextRank
LexRank
Lsa
Luhn
KL

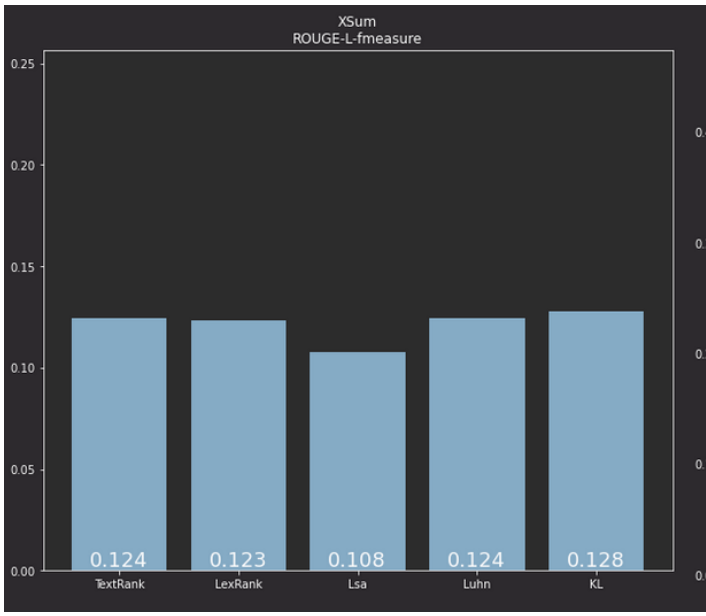

XSum
BLEU-2
0.25
0.20
0.15
0.10
0.05
0.00
0.047
0.051
0.043
0.048
0.053
TextRank
LexRank
Lsa
Luhn
KL

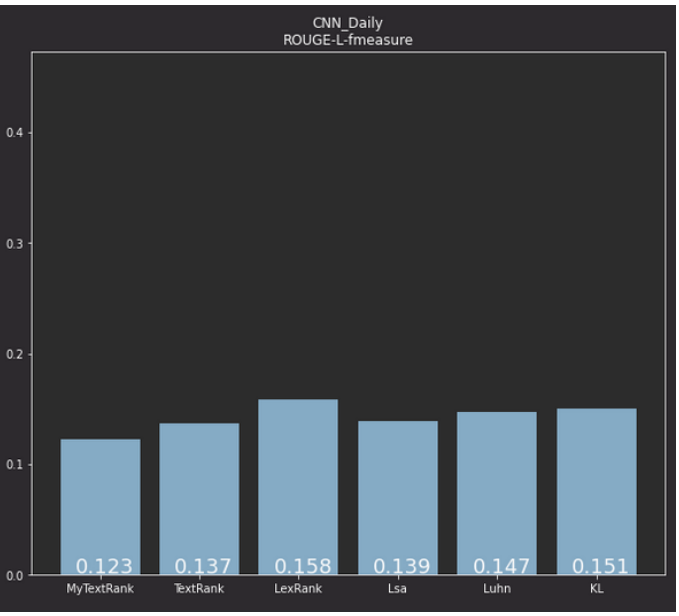
CNN_Daily
BLEU-2
0.4
0.3
0.2
0.1
0.0
0.057
0.067
0.086
0.074
0.074
0.080
MyTextRank
TextRank
LexRank
Lsa
Luhn
KL











Best model on average

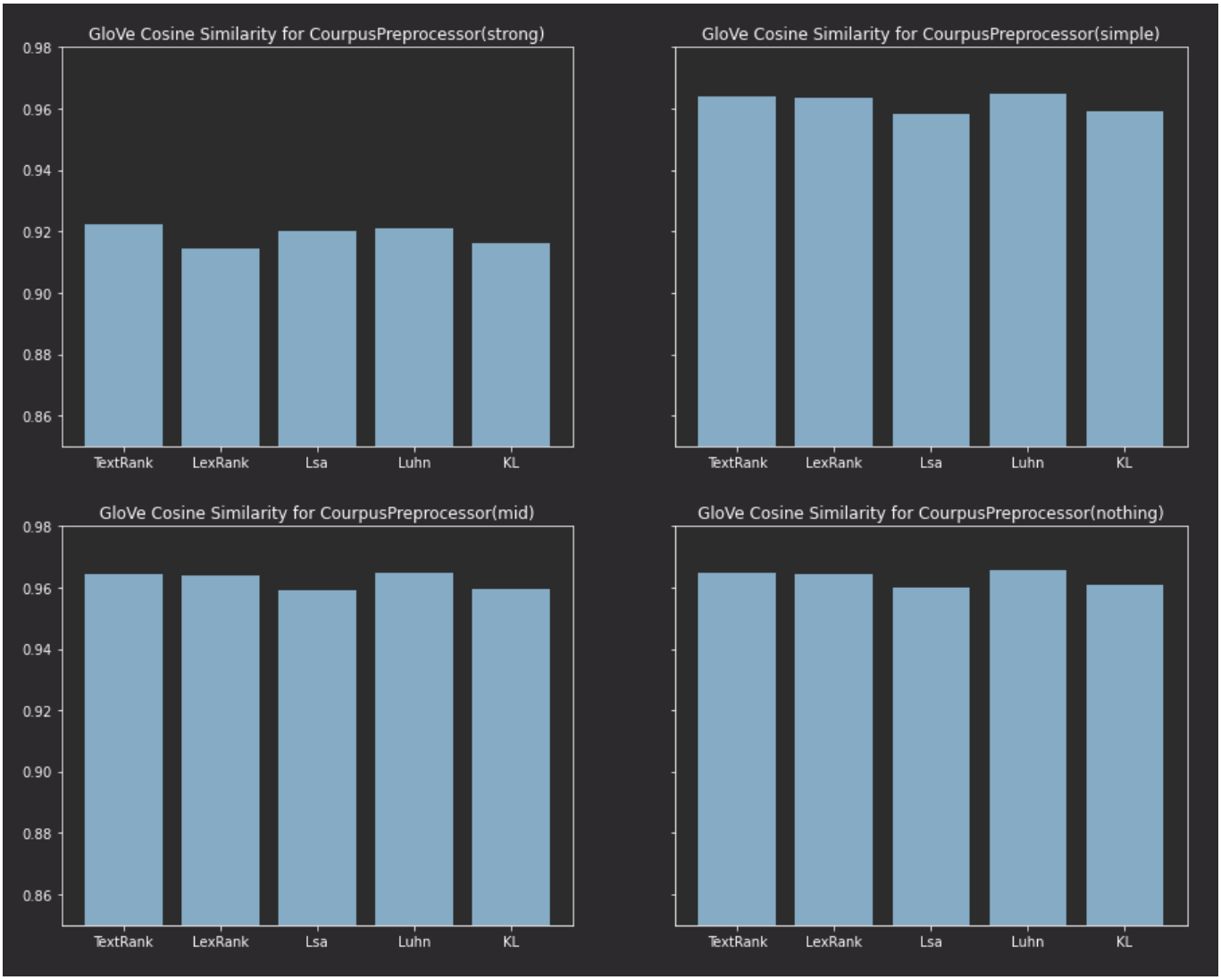
```
// XSum
{'BLEU-1': 'LexRank',
 'BLEU-2': 'KL',
 'BLEU-3': 'KL',
 'BLEU-4': 'KL',
 'ROUGE-1-fmeasure': 'Luhn',
 'ROUGE-1-recall': 'TextRank',
 'ROUGE-2-fmeasure': 'KL',
 'ROUGE-2-recall': 'TextRank',
 'ROUGE-L-fmeasure': 'KL',
 'ROUGE-L-recall': 'TextRank'}
// CNN_Daily
{'BLEU-1': 'LexRank',
 'BLEU-2': 'LexRank',
 'BLEU-3': 'LexRank',
 'BLEU-4': 'LexRank',
 'ROUGE-1-fmeasure': 'LexRank',
 'ROUGE-1-recall': 'Luhn',
 'ROUGE-2-fmeasure': 'LexRank',
 'ROUGE-2-recall': 'Luhn',
 'ROUGE-L-fmeasure': 'LexRank',
 'ROUGE-L-recall': 'Luhn'}
```

Best preprocessing

```
// XSum
{'BLEU-1': 'CourpusPreprocessor(mid)',
 'BLEU-2': 'CourpusPreprocessor(mid)',
 'BLEU-3': 'CourpusPreprocessor(mid)',
 'BLEU-4': 'CourpusPreprocessor(mid)',
 'ROUGE-1-fmeasure': 'CourpusPreprocessor(nothing)',
 'ROUGE-1-recall': 'CourpusPreprocessor(simple)',
 'ROUGE-2-fmeasure': 'CourpusPreprocessor(nothing)',
 'ROUGE-2-recall': 'CourpusPreprocessor(simple)',
 'ROUGE-L-fmeasure': 'CourpusPreprocessor(mid)',
 'ROUGE-L-recall': 'CourpusPreprocessor(simple)'}
// CNN_Daily
{'BLEU-1': 'CourpusPreprocessor(mid)',
 'BLEU-2': 'CourpusPreprocessor(mid)',
 'BLEU-3': 'CourpusPreprocessor(mid)',
 'BLEU-4': 'CourpusPreprocessor(simple)',
 'ROUGE-1-fmeasure': 'CourpusPreprocessor(nothing)',
 'ROUGE-1-recall': 'CourpusPreprocessor(nothing)',
 'ROUGE-2-fmeasure': 'CourpusPreprocessor(nothing)',
```

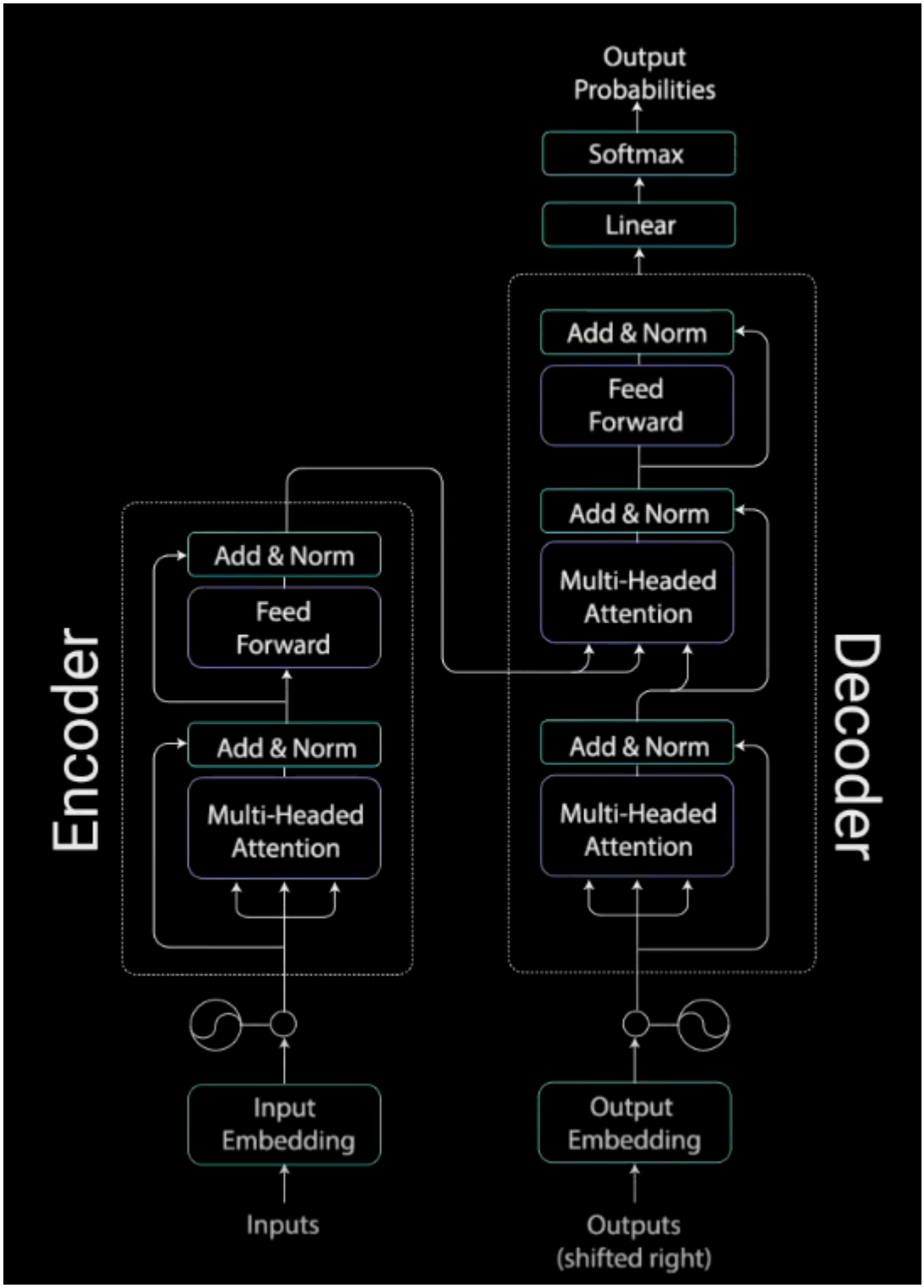
```
 'ROUGE-2-recall': 'CourpusPreprocessor(nothing)',
 'ROUGE-L-fmeasure': 'CourpusPreprocessor(nothing)',
 'ROUGE-L-recall': 'CourpusPreprocessor(nothing)'}
```

GloVe cosine similarity

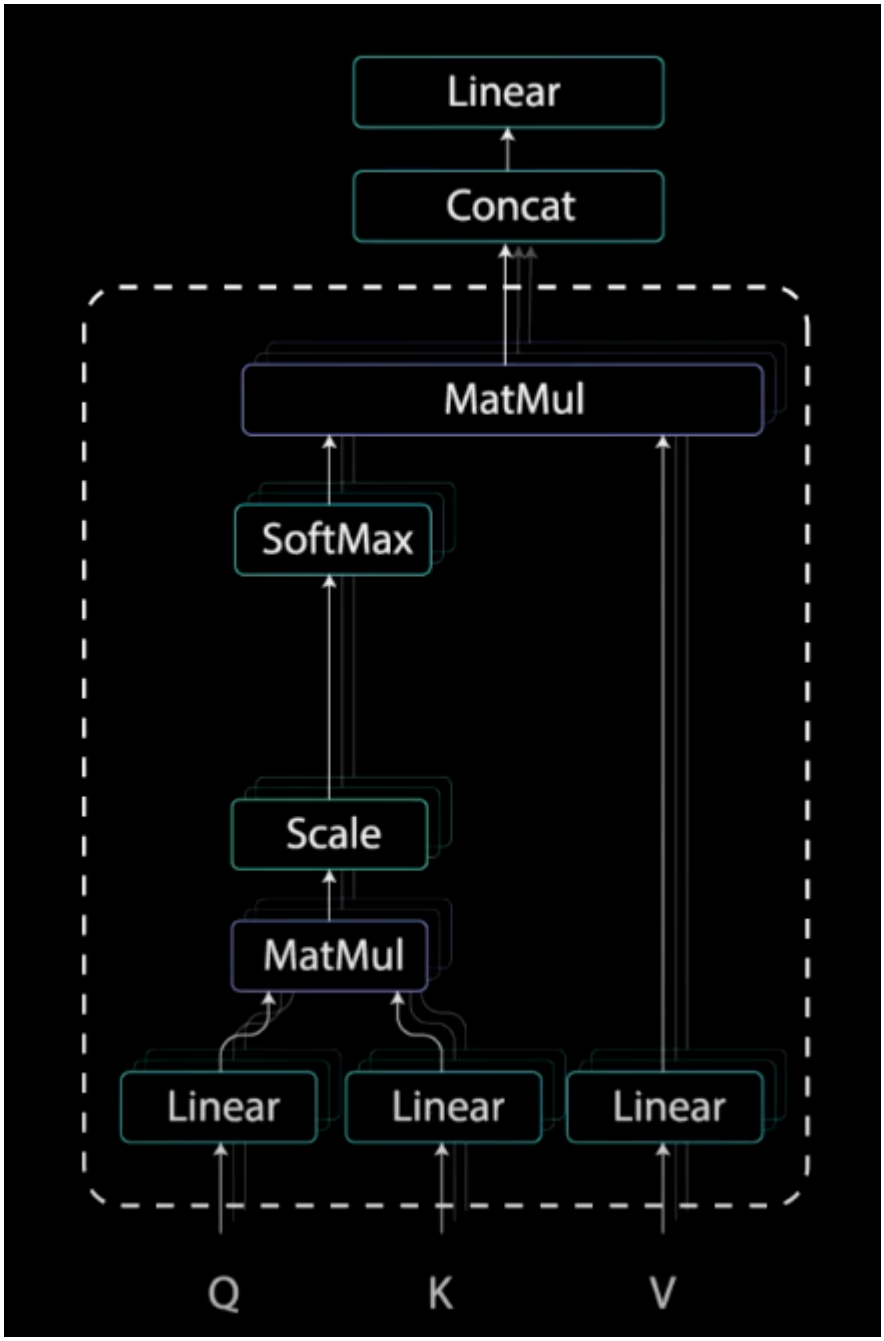


Transformers

Encoder-Decoder

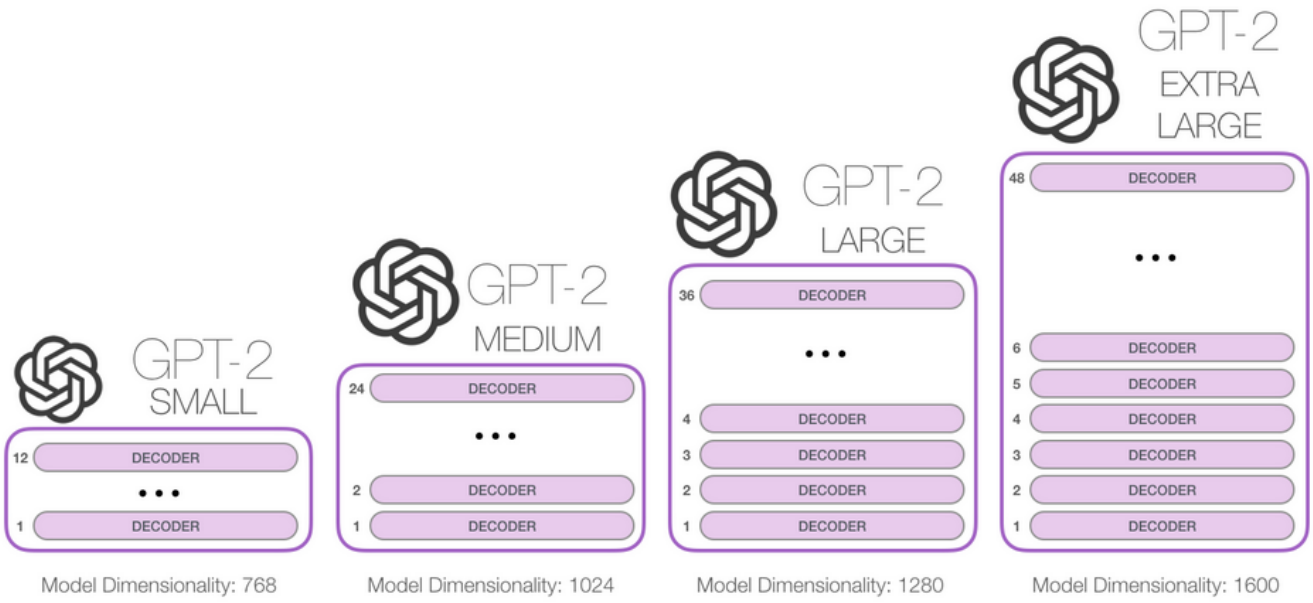


Self attention



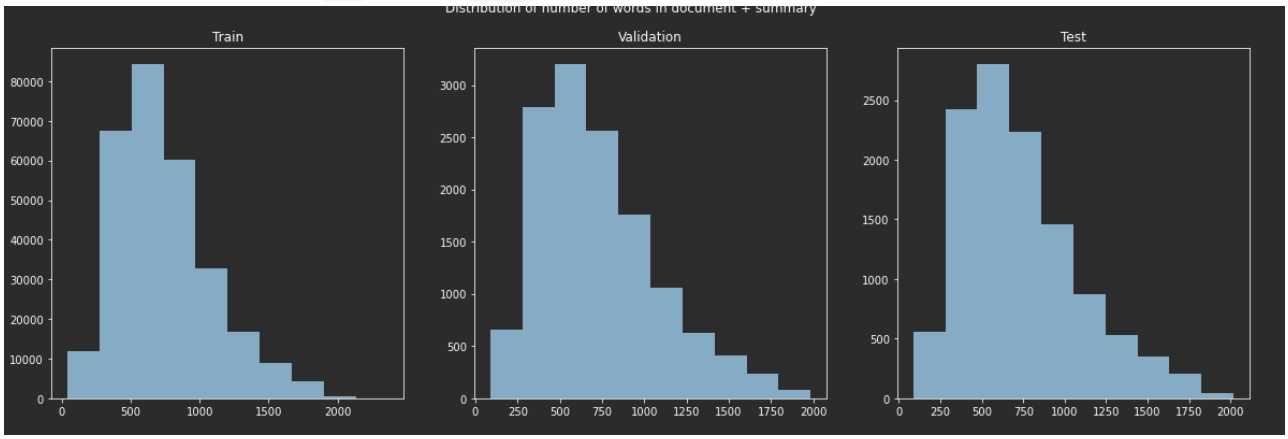
GPT-2 training

- `gpt2 small` - 117M parameters
- Vocabulary size ~= 50K
- Context size 1024
- Embedings size 768
- Masked self attention
- Auto-regression



How to train

- Tokenize
- input - "article `<sep>` summary `<pad><pad>...<pad>`"
- limited context size (`gpt` - 512, `gpt2` - 1024)



Where to train

- **Google cloud**
- free 300 bucks
- 16GB Tesla T4

How to use

Decoding strategy

- Top-K
- With temperature

$$P(x_i|x_{1:i-1}) = \frac{exp(u_i/t)}{\sum_j exp(u_j/t)}$$

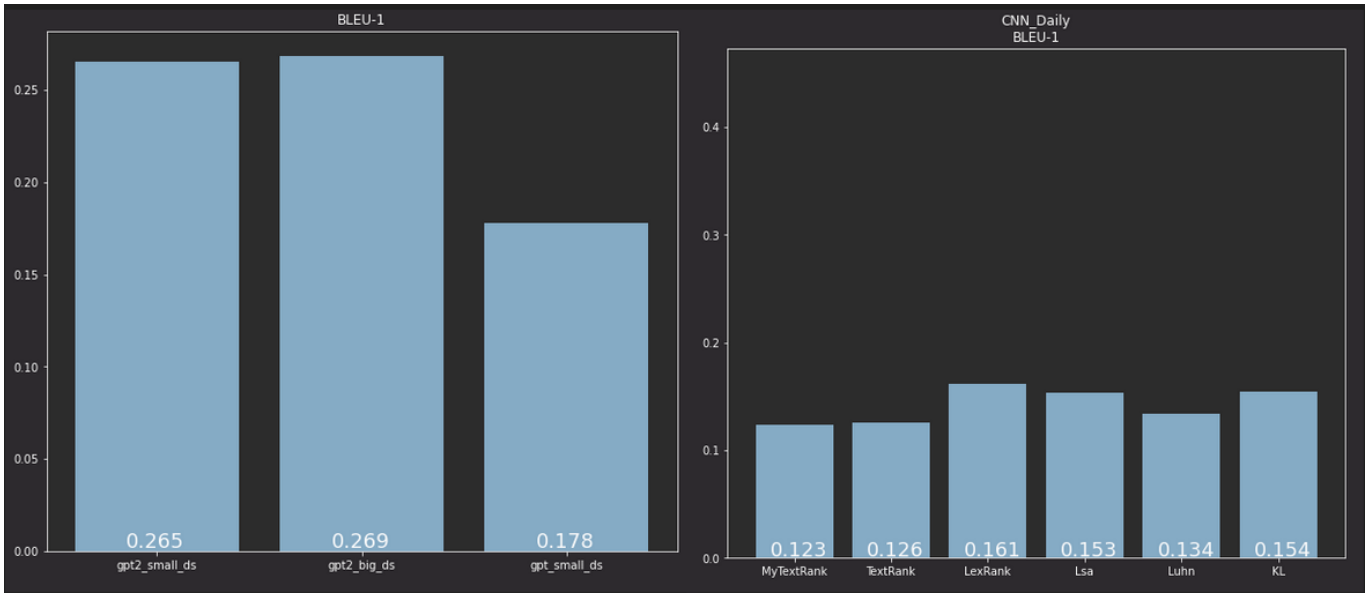
- Nucleus

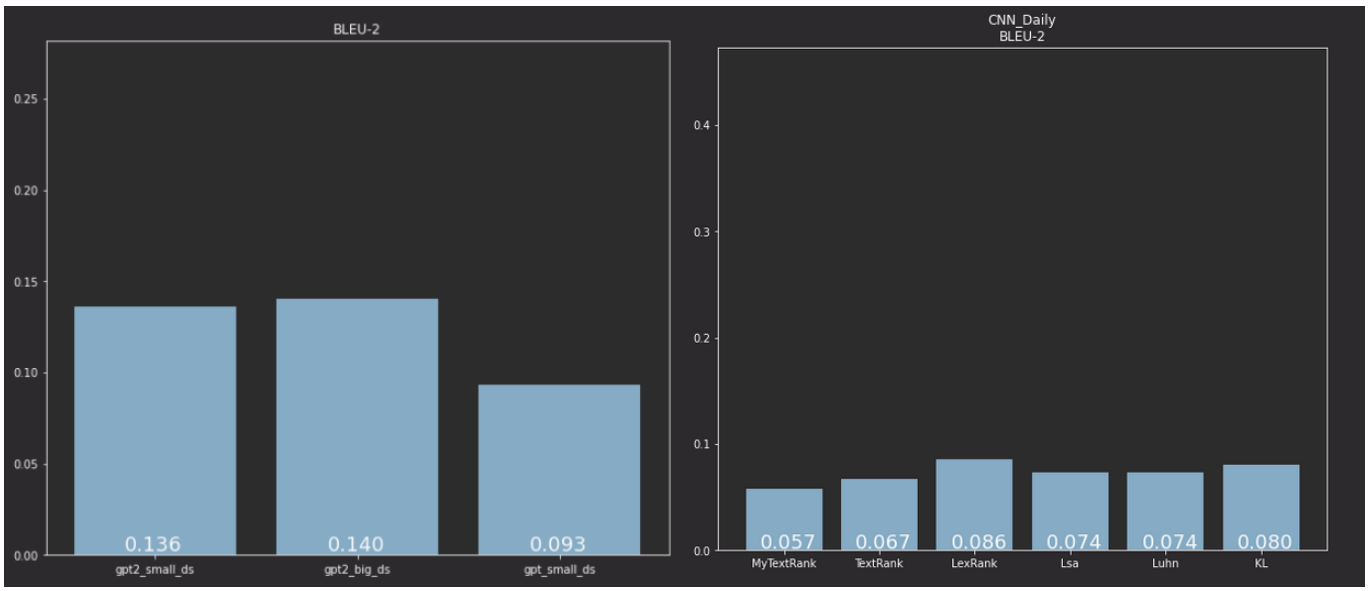
$$\sum_{x \in V_p} P(x_i|x_{1:i-1}) >= p$$

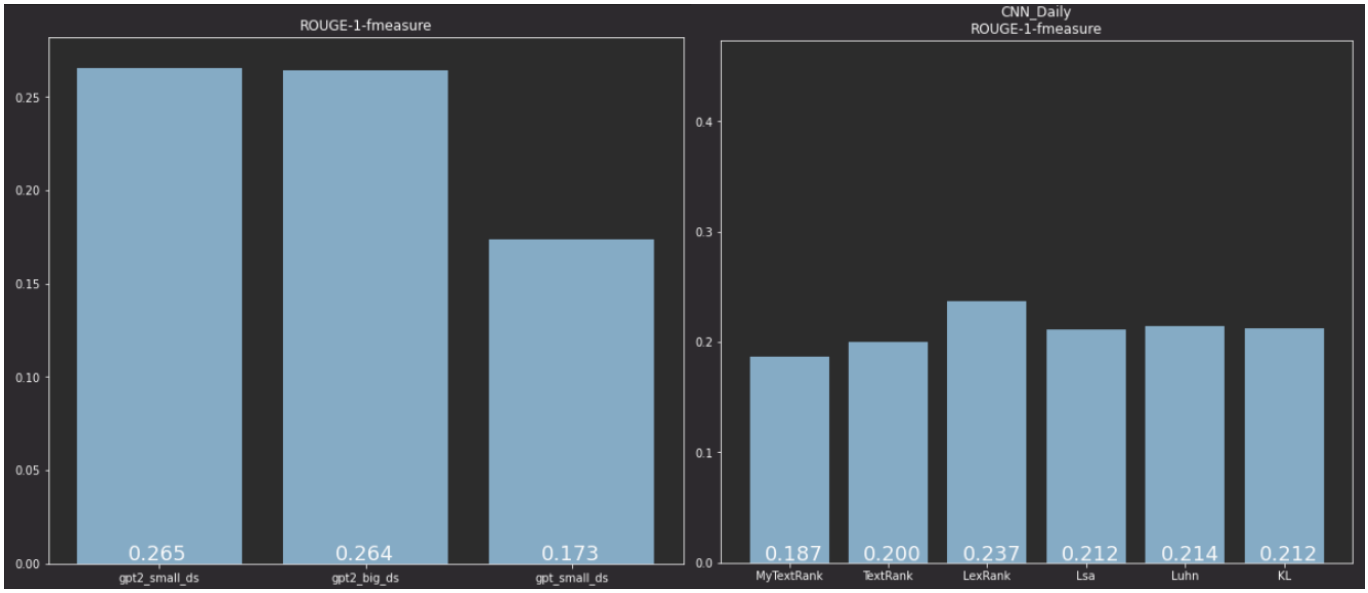
- `top_k` ~= 20
- `temperature` ~= 0.8
- `top_p` ~= 0.8

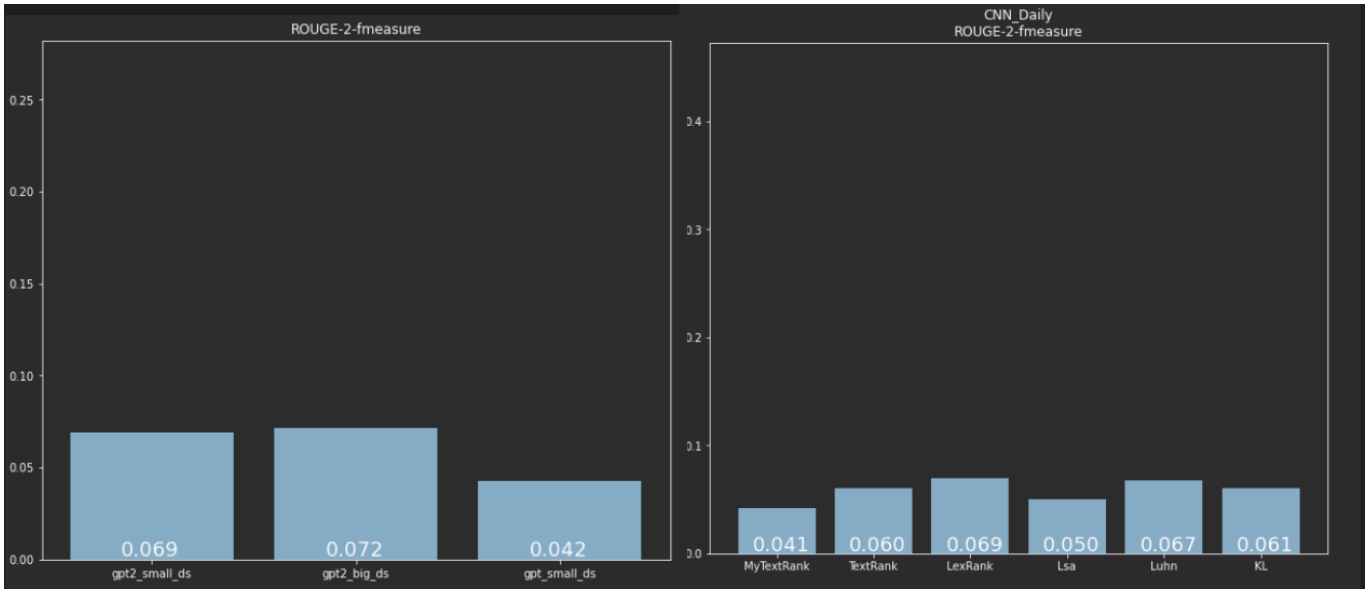
Results

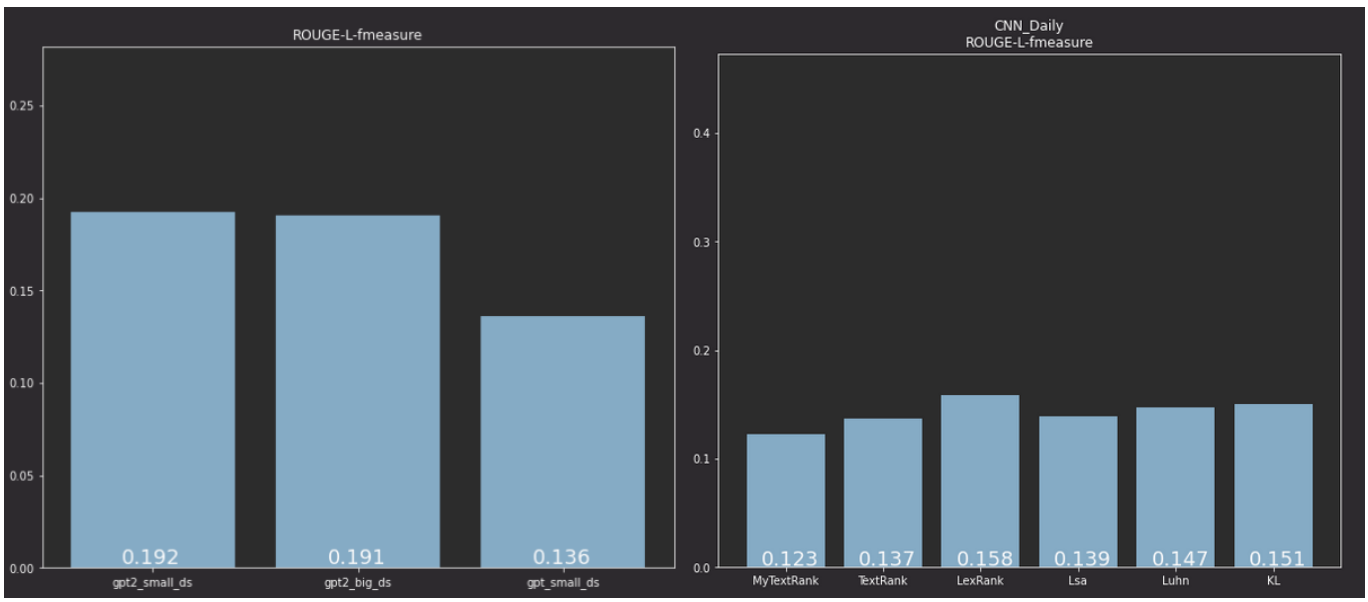
small_ds - 10k
big_ds - 50K






ROUGE-1-fmeasure
0.265
0.264
0.173
gpt2_small_ds
gpt2_big_ds
gpt_small_ds
CNN_Daily
ROUGE-1-fmeasure
0.187
0.200
0.237
0.212
0.214
0.212
MyTextRank
TextRank
LexRank
Lsa
Luhn
KL


ROUGE-2-fmeasure
0.069
0.072
0.042
gpt2_small_ds
gpt2_big_ds
gpt_small_ds
CNN_Daily
ROUGE-2-fmeasure
0.041
0.060
0.069
0.050
0.067
0.061
MyTextRank
TextRank
LexRank
Lsa
Luhn
KL


ROUGE-L-fmeasure
0.192
0.191
0.136
gpt2_small_ds
gpt2_big_ds
gpt_small_ds
CNN_Daily
ROUGE-L-fmeasure
0.123
0.137
0.158
0.139
0.147
0.151
MyTextRank
TextRank
LexRank
Lsa
Luhn
KL

Example

```
"""
==================================================
Actual summary:

Chelsea beat Stoke 2-1 in Premier League at Stamford Bridge on Saturday.
Cesc Fabregas' nose was left bloodied after tussle with Charlie Adam.
Midfielder showed off injury as he posted picture to Instagram after game.
Fabregas joked that he might finally be able to get his nose fixed.
==================================================
GPT-2 summary:

Spain's Fabregas has suffered a nasty-looking nose during the match .
The former Barcelona midfielder was left bloodied after being caught by a
flailing arm .
The match left Spain's Fabregas with a bloody nose and required treatment .
The 27-year-old has suffered a broken nose in the past but
==================================================
LexRank summary:

Cesc Fabregas showed off the result of a nasty-looking clash with Charlie
Adam's arm during Chelsea's victory against Stoke City at Stamford Bridge on
Saturday.
The former Barcelona midfielder was left bloodied after being caught by a
flailing arm following a tussle with Adam, the scorer of a 66-yard wonder
goal.
Spain international Fabregas also took a whack on his leg during the clash on
Saturday .
He avoided having an operation on the injury then, but it seems that Fabregas
might be happy to have corrective surgery this time.
"""
```

References

[1] - https://huggingface.co/tasks/summarization
[2] - https://www.analyticsvidhya.com/blog/2018/11/introduction-text-summarization-textrank-python/
[3] - Illustrated Guide to Transformers- Step by Step Explanation
[4] - The Illustrated GPT-2 (Visualizing Transformer Language Models)
[5] - Generating Text Summaries Using GPT-2 on PyTorch with Minimal Training
[6] - Decoding Strategies that You Need to Know for Response Generation