Abstractive Text Summarization on CNN/DailyMail Dataset Using T5 Transfer Learning Model

Aziza Mirsaidova, Vinit Todai, Shalin Parikh
Northwestern University

Abstract

This paper discusses ***abstractive text summarization*** which summarizes the text maintaining coherent information in a similar amount of words as human generated summary. We briefly describe the abstractive text summarization task and several methods used to predict the summary in a concise way. Namely, we fine tune ***T5 transfer learning*** model on ***CNN/DailyMail dataset*** and achieve a strong ***ROUGE-1 unigram*** measure of ***44%*** and ***ROUGE-2 bigram*** of ***14%***.

1. Introduction

One of the main tasks of information retrieval in Natural Language Processing (NLP) is the ability to extract information in a coherent and concise manner. With the help of advances in language models, now we can develop text summarization models, which is a "NLP tasks that automatically converts a text, or a collection of texts within the same topic, into a concise summary that contains key semantic information which can be beneficial for many downstream applications such as creating news digests, search engine, and report generation."

Summarization is the process of condensing a long piece of text into a shorter version that retains the essential information. There are two types of summarization methods: extractive and abstractive. Extractive methods create summaries solely from passages (usually whole sentences) extracted directly from the source text, whereas abstractive methods may generate new words and phrases not found in the source text – as a human-written abstract usually does.

To be able to automatically generate the summary that is both succinct and coherent, we will be utilizing Abstractive Text Summarization.

2. Literature Review

Because copying huge portions of text from the original material provides baseline levels of grammaticality and correctness, the extractive technique is simpler. On the other hand, sophisticated abilities like paraphrasing, generalization, and the incorporation of real-world information are only conceivable in an abstractive framework.

Because abstractive summarization is difficult, the vast majority of previous work has been extractive (Kupiec et al., 1995; Paice, 1990; Saggion and Poibeau, 2013). Abstractive summarization is now possible because to the recent success of sequence-to-sequence models (Sutskever et al., 2014), in which recurrent neural networks (RNNs) both read and freely produce text (Chopra et al., 2016; Nallapati et al., 2016; Rush et al., 2015; Zeng et al., 2016). Though these systems are promising, they display undesired characteristics such as imprecise factual reproduction, inability to cope with out-of-vocabulary (OOV) terms, and the tendency to repeat themselves.

Christopher D. Manning, Abigail See, Peter J. Liu Get to the Point: Summarization with Pointer-Generator Networks describes an architecture for multi-sentence summaries that overcomes these three concerns. While much previous abstractive work has focused on headline generation problems (reducing one or two phrases to a single headline), the authors argue that summarizing lengthier texts is both more difficult (requiring greater degrees of abstraction while avoiding repetition) and ultimately more beneficial.

Augmenting the standard sequence to sequence model, See and his colleagues proposed a pointer generator model with coverage mechanism that outperformed the current state of the art model by 2 ROUGE points. Initially, they applied a pointer hybrid network for pointing which copies the source text and retains the accurate representation of the information, retaining the ability to produce a word via generator.

They tested their model on the CNN/ Daily Mail dataset (Hermann et al., 2015; Nallapati et al., 2016), which comprises news items (on average 39 lines) linked with multi-sentence summaries, and found that they outperformed the state-of-the-art abstractive system by at least 2 ROUGE points.

The paper's hybrid pointer-generator network supports copying words from the source text by pointing (Vinyals et al., 2015), which enhances OOV word correctness and handling while keeping the capacity to produce new words (Vinyals et al., 2015).

The network is comparable to Gu et al(2016) .'s CopyNet and Miao and Blunsom's (2016) Forced-Attention Sentence Compression, both of which were used for short-text summarization.

The research introduces a new form of the Neural Machine Translation coverage vector (Tu et al., 2016), which is used to track and regulate coverage of the source content. The goal of the study is to demonstrate that coverage is a powerful tool for avoiding duplication.

3. Dataset

CNN/DailyMail. Hermann and his colleagues addressed the lack of real natural language training data by developing a "supervised reading comprehension data set" using a new method. They discovered that utilizing entity detection and anonymisation methods, a summary of the phrases and their related dataset may be turned to "context-query-answer" triples. The authors used this strategy to construct two machine-readable corpora from online newspaper articles and their summaries. The authors gathered 93,000 articles from CNN and 220,000 articles from DailyMail, both of which give a summary of the information in this dataset.

	CNN			Daily Mail		
	train	valid	test	train	valid	test
# months	95	1	1	56	1	1
# documents	90,266	1,220	1,093	196,961	12,148	10,397
# queries	380,298	3,924	3,198	879,450	64,835	53,182
Max # entities	527	187	396	371	232	245
Avg # entities	26.4	26.5	24.5	26.5	25.5	26.0
Avg # tokens	762	763	716	813	774	780
Vocab size	118,497			208,045		

Table 1: Summary statistics of corpus given by Herman et. al (2015).

The statistics of the corpus are summarized in Table 1. There are 287,113 data for training, 13,368 for validation, and 11,490 for testing in this dataset. The training set contains an average of 28 sentences per document.

Data pre-processing. Starting with the project, first we loaded the CNN_DailyMail dataset from the hugging face datasets. The dataset consists of three features namely id, article and highlights.

	target_text	source_text
0	Bishop John Folda, of North Dakota, is taking ...	summarize: By . Associated Press . PUBLISHED: ...
1	Criminal complaint: Cop used his role to help ...	summarize: (CNN) -- Ralph Mata was an internal...
2	Craig Eccleston-Todd, 27, had drunk at least t...	summarize: A drunk driver who killed a young w...
3	Nina dos Santos says Europe must be ready to a...	summarize: (CNN) -- With a breezy sweep of his...
4	Fleetwood top of League One after 2-0 win at S...	summarize: Fleetwood are the only team still t...

Table 2: The dataset consists of three features namely id, article and highlights.

The id column represents the id of an article, article column gives the heading and the content of the whole news piece and the highlights column gives a brief summary of the article mentioned in the previous column. Average number of tokens for articles is 781, and summary is 56 tokens. The dataset is split into train, validation, and test sets. We are utilizing only the train set which consists of 287,113 rows of news articles.

4. Methodology

Sequence-to-sequence Modeling. To construct the summary, abstractive summarizers do not use sentences from the original text passage. Instead, they create a paraphrase of the given text's major points, using a vocabulary set that differs from the original. Sequence-to-sequence (Seq2Seq) modeling can be used to comprehend abstractive text summarization since it contains sequential information. Because the models are designed to construct an output sequence of words from an input sequence of words, they are referred to as "sequence to sequence models." The actual text document is the input sequence in this case, and the abbreviated summary is the output sequence. According to our findings, the major components of a Seq2Seq model are Encoder and Decoder Networks.

Encoder and Decoder Networks

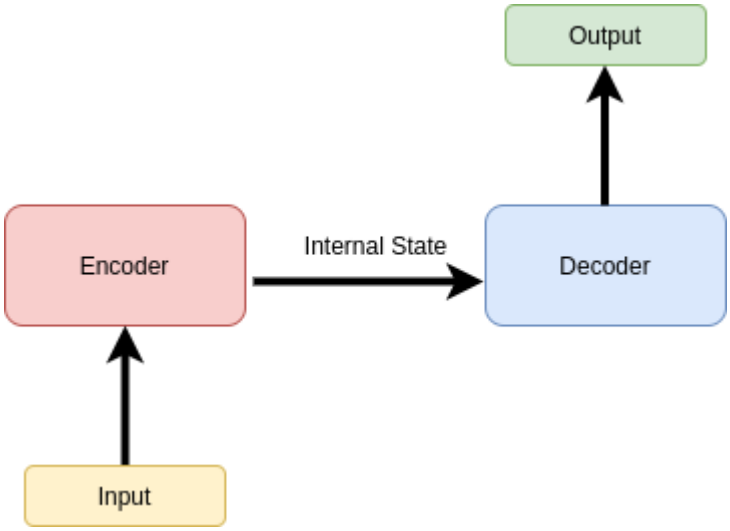

Figure 1: The diagram above depicts the model's basic architecture.

The encoder's job is to take the input text or document and turn it into a final state vector (hidden state and cell state). The internal state in the diagram represents this. LSTM, RNN, and GRU layers may be present in the encoder. Because of the removed Exploding and Vanishing Gradient problem, mostly LSTM layers are utilized.

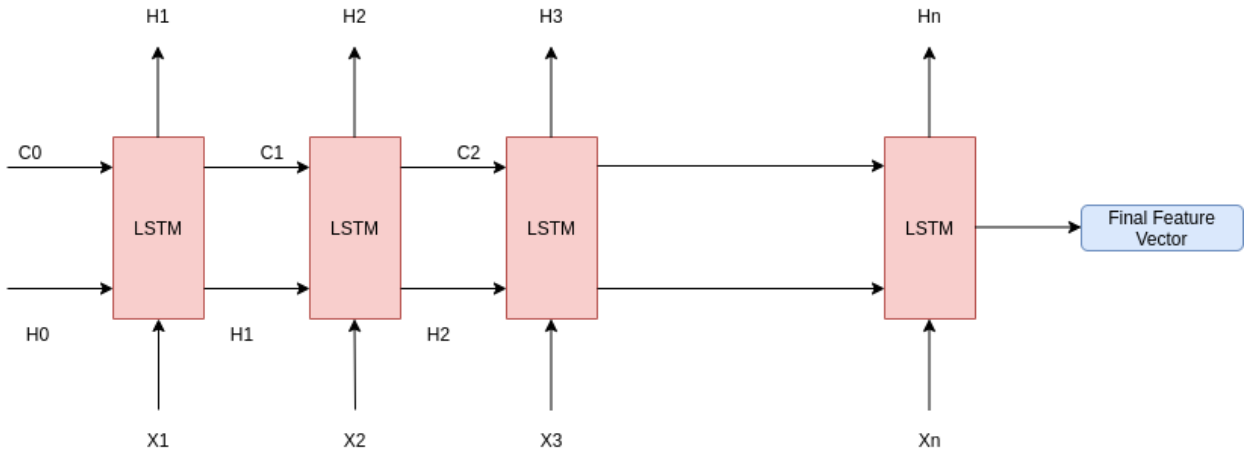

Figure 2: The above diagram shows how the encoder network will look like.

One word is supplied to the encoder network at a time step, and the hidden state and cell states constitute our final state or feature vector after the nth input word is given to the LSTM layer.

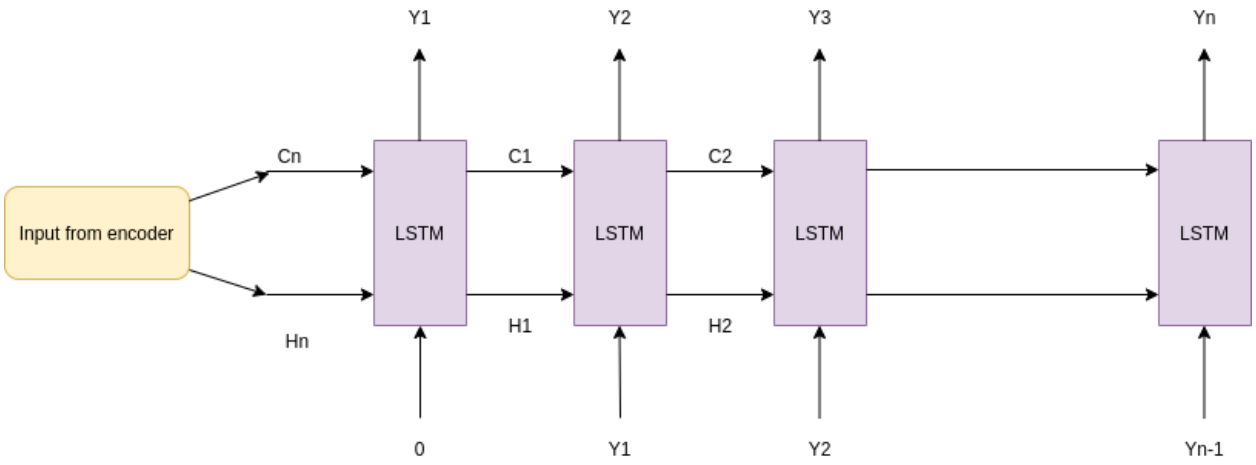

Figure 3: Design of Decoder Layer

The inputs from the encoder's final states, which are the hidden and cell state activations, are now sent to the first layer. Given the previous word formed, the decoder model takes the inputs and creates the predicted words in the output sequence.

The encoder-decoder model is trained on a set of words called a target set or vocabulary. The LSTM's hidden activation is now transmitted via a softmax layer in each step of the decoder, which calculates the probability for each word in the lexicon to be predicted as the next word. At that time step, the word with the highest probability is picked as the output. The model uses a dataset to train and converts the problem to a supervised classification problem.

Extraction of keywords. After vectorization, word embeddings are employed to represent the words in the texts. However, they are insufficient since, in order to summarize, we must concentrate on the context and keywords in the text. To represent a word, we evaluate characteristics such as part of speech tags, named-entity tags, and TF IDF statistics, as well as embeddings. Using bins, we transform continuous TF IDF data to categorical values. Finally, we take all of a word's characteristics and embeddings and combine them into a new embedding.

So, the TFIDF and POS tags tell us how essential the words are in the context of the document, and the word embeddings offer us a basic notion of the term. Following that, we combine them into a single long vector and feed them to the network.

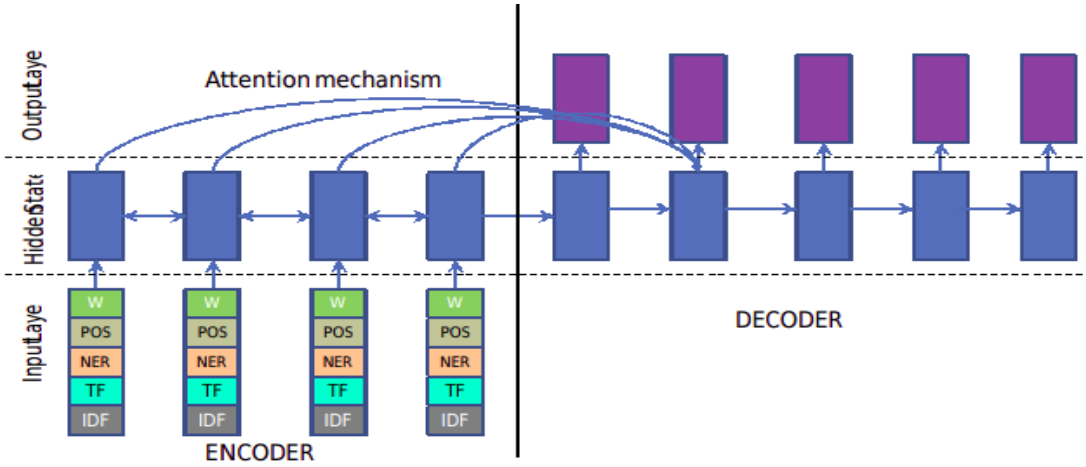

Figure 4: Architecture of Encoder and Decoder layer for text summarization.

In the beginning of our modeling process, we worked on building Seq-to-Seq model using Tensorflow applying pre-processing as TF-IDF, removing stopwords, tag words, identifying vocabulary and integrating ConceptNet with utilizing RNN on the encoder and LSTM on the decoder. However, due to challenges with computational complexities and time constraints, we decided to use the pre-trained T5 model described in the following section.

T5 model. The T5 (Text-to-Text Transfer Transformer) model was developed as a result of a large-scale study (Raffel et. al., 2019) that looked at the boundaries of transfer learning. It is based on prominent designs such as GPT, BERT, and RoBERTa (to mention a few) that have had great success with Transfer Learning. While BERT-like models may be fine-tuned to accomplish a number of tasks, the architecture's restrictions limit each model to just one.

This is often accomplished by superimposing a task-specific layer on top of the Transformer model. A BERT Transformer, for example, may be modified to do binary classification by adding a fully-connected layer with two output neurons (corresponding to each class). All NLP tasks are reframed as text-to-text tasks in

the T5 paradigm, which breaks from precedent. As the model's input and output are always strings, this results in a shared framework for every NLP activity. In the case of binary classification, the T5 model will simply output the class as a string (i.e. "0" or "1").

Because the input and output formats for each NLP job are identical, a single T5 model may be trained to execute many tasks. We may simply prepend a prefix (string) to the model's input to define which job should be done. This notion is demonstrated in the image (seen below) from the Google AI Blog post.

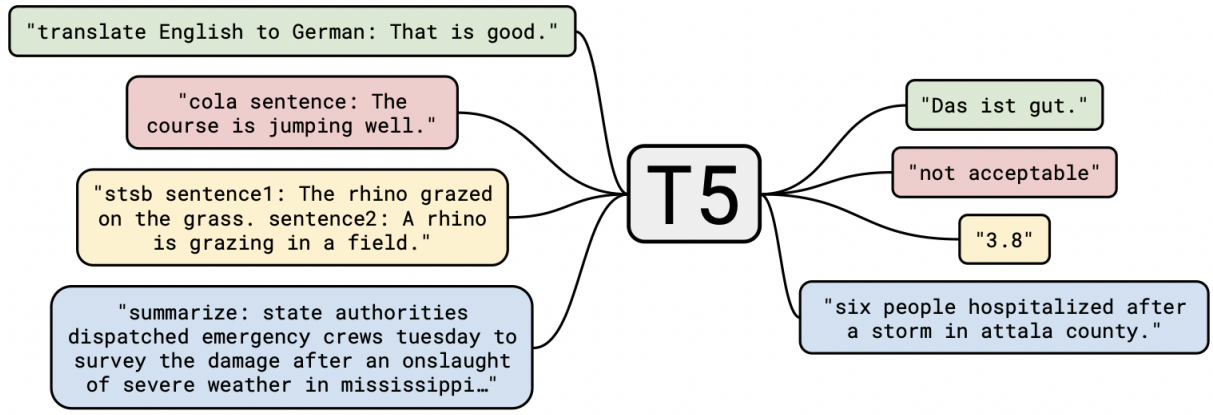

Figure 5: T5 conducted multiple NLP tasks such as question-answering, translation, text generation, text summarization and more.

We fine tuned the T5 model with utilizing following parameters:

- Maximum length of sentences: 512
- Length of the summary text: 120
- Training size: 5000
- Epoch size: 10
- Batch size: 4

At first, we trained using batch size 8. However due to the computation complexities and space constraints while utilizing GPU, we decided to use the batch size of 4 and our results are explained in the below section.

5. Evaluation Metrics.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation). ROUGE measures to compare our extracted summaries to the gold standard ones. ROUGE measures are a collection of metrics that apply to automatic text summarization and machine translation.

ROUGE measures are used to compare the produced summary to a (gold standard) man-made reference summary, with EQUATION for recall and EQUATION for accuracy. Study conducted by Lin on the ROUGE score thoroughly explains the benefits and drawbacks of Rouge in all of its forms. Across (our programmed) system summary and reference summary, ROUGE-1 and ROUGE-2 quantify overlapping unigrams and bigrams, respectively.

For brief summary tasks (i.e. headlines), Rouge-2 does not have a good Pearson correlation with human vs human judgments, as seen in the image. Rouge-1 is typically superior.

We used ROGUE unigram and bigram scores to measure the performance of our model, as the vast majority of relevant literature recommends. The following formulae were used to determine recall and precision:

$$Recall = \frac{ngrams_{system} \bigcap ngrams_{reference}}{ngrams_{reference}} \text{ and } Precision = \frac{ngrams_{system} \bigcap ngrams_{reference}}{ngrams_{system}}$$

With extractive approaches, the summary against which we would assess would almost certainly contain different terms, limiting the ROUGE recall value; for example, if the reference contains 15% of words not found in the original text, the highest potential recall value is 85 %.

6. Results

What is interesting about text summarization, is that a great part of the qualitative analysis will be placed on whether the summary is humanly readable and contains sufficient information. We have taken care of it being concise by limiting its size.

Original Summary	Predicted Summary	Rouge 1 Score	Rouge 2 Score
Alan Dodd painted the stunning and unusual murals on the side of a friend's home in Eye, Suffolk . He took 55 hours to complete the work on the 50-foot-long house side and says it is a tribute to the town. He paints historical scenes and was taught by influential artist David Hockney.	Alan Dodd, 71, spent 55 hours up a ladder to paint the stunning mural on the side of a friend's home in Eye, Suffolk. He says it is a homage to the town's history and will benefit from being on a wall that stays out of the sun.	R:0.878 P:0.109 F:0.194	R:0.44 P:0.04 F:0.07
Joshua Sadler, 21, lost control on 60mph country road and hit a tree. Front seat passenger Mikey Maguire, 19, was killed in the crash. Sadler had previous convictions for a string of serious motoring offences. He was twice banned from the road before he even had a licence. Seen on the night of the fatal crash performing handbrake turns in a car park. Victim's mother described the 12 month sentence as an 'insult' to her son's life	Joshua Sadler, 21, lost control on 60mph country road and smashed into a tree. Front seat passenger Mikey Maguire, 19, was killed in the crash. Sadler had previously been disqualified from driving for two counts of aggravated vehicle taking. He was twice banned from the road. before he even had a license. Victim's mother described 12 month sentence as an 'insult' to her son's life.	R:0.904 P:0.214 F:0.346	R:0.648 P:0.105 F:0.181
Swearing peaks at 12 a.m. to 1:30 a.m. Top seven curses accounted for over 90% of offensive tweets. one-month sample of 51 million English-language tweets analysed.	One-month sample of 51 million English-language tweets were analysed. Top seven curses accounted for over 90% of offensive tweets. curses rise from 12 a.m. to 1:30 a.m.	R:0.833 P:0.061 F:0.115	R:0.730 P:0.033 F:0.064

	before bedtime.		
The dogs can lose their sensing abilities if overworked. FEMA dogs that arrived more recently will continue working. The death toll for the tragedy now stands at 21 with 30 still missing. Another four bodies were uncovered Sunday but cannot be added to the official list of victims until they are identified.	Many of the exhausted dogs that have been in search of victims of the massive mudslide will take a two-day break. Dogs from the Federal Emergency Management Agency that arrived more recently will continue working. The size of the debris field is also smaller than originally thought.	R:0.877 P:0.075 F:0.138	R:0.490 P:0.025 F:0.048
Police were pursuing a sedan with four people following reports of alleged robbery at Macedon railway station, northwest of Melbourne. The car was found a short time later on its side early on Saturday morning. Police found one man dead after being ejected from the vehicle . A 27-year-old woman was arrested at scene and a man, 30, was arrested later that morning on a roadside. Police were still looking for another man when he handed himself in at Gisborne Police Station on Saturday night.	A man has handed himself in almost 12 hours after allegedly running from the scene of a fatal car accident northwest of Melbourne. Police were responding to reports of an alleged robbery at Macedon railway station, about 70km from Melbourne, just before 5am on Saturday. The four people took off in a sedan and police gave chase but a short time later, officers found the car had rolled on a nearby bend. A 27-year-old woman was arrested at the scene and a man, 30, was later arrested on a roadside at nearby Gisborne Police Station. A 32-year-old from Diggers Rest showed up at Gisborne Police Station when he was arrested.	R:0.857 P:0.295 F:0.439	R:0.450 P:0.120 F:0.194

Table 3: Results of Fine tuned T5 model with corresponding original and predicted summary and rouge 1, rouge 2 scores.

In the above table, the yellow and the blue highlights indicate the overlap between the original summary and our predicted summary. We can see that our model has

generated nearly the same summaries as provided by the author and in some cases, the model has even included more relevant data from the original paragraph.

The fact that the sentences are often quite long appears to be evident. Some of our errors may be due to a mix of this and our failure to completely extract sentences similar to the goldfile when the articles do not always contain the same words and phrases as the gold summary, even though the same ideas are conveyed in both. This points to a difficulty with ROUGE as a metric in and of itself—perhaps a system that takes semantic similarity into consideration might be more suited to the challenge, despite the fact that ROUGE is a more widely used metric in the present literature.

7. Future Work

Abstractive text summarization is one of the challenging tasks of text summarization. In order to summarize textual data well, language models need to be able to comprehend documents and extract the relevant information which are highly challenging for computers. As we have experienced while working with vanilla Seq-to-Seq models, when the length of a document increases, so does the complexity of the model. For future work, we plan to train the CNN/DailyMail Dataset using current state of the art abstractive text summarization model - PEGASUS in which the “important sentences are removed/masked from an input document and are generated together as one output sequence from the remaining sentences, similar to an extractive summary” as said in paper presented by Zhang and his colleagues in 2020.

References

1. See, A., Liu, P. J., & Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. ***arXiv preprint arXiv:1704.04368***. https://arxiv.org/pdf/1704.04368.pdf
2. Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., & Blunsom, P. (2015). Teaching machines to read and comprehend. ***Advances in neural information processing systems***, ***28***.
3. Dima Suleiman, Arafat Awajan (2015). Deep Learning Based Abstractive Text Summarization: Approaches, Datasets, Evaluation Measures, and Challenges
4. ***Jon Deaton, Austin Jacobs, Kathleen Kenealy, Abigail See: Transformers and Pointer-Generator Networks for Abstractive Summarization***
5. Priya D. 2019. Text Summarization using Deep Learning. Medium: Towards Data Science. Web: https://towardsdatascience.com/text-summarization-using-deep-learning-6e379ed2e89c
6. DeepMind Q&A Dataset. https://cs.nyu.edu/~kcho/DMQA/
7. Lin, 2004. ROUGE: A Package for Automatic Evaluation of Summaries. http://www.aclweb.org/anthology/W04-1013
8. Pegasus Results. https://github.com/google-research/pegasus
9. Zhang, J., Zhao, Y., Saleh, M., & Liu, P. (2020, November). Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In International ***Conference on Machine Learning*** (pp. 11328-11339). PMLR.
10. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... & Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. ***arXiv preprint arXiv:1910.10683***.