International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 09 | Sep 2021 www.irjet.net p-ISSN: 2395-0072

Automated Text Summarization of News Articles

Varun Deokar[1], Kanishk Shah[2]

[1]*Student, Dept. of Information Technology, Vidyalankar Institute of Technology, Maharashtra, India*
[2]*Student, Dept. of Computer Science, D.J. Sanghvi College of Engineering, Maharashtra, India*

Abstract - *This paper focuses on a technique to automatically summarize news articles from the web by just typing in keywords related to your article. It uses the transformers model and compares the Bart and T5 model to see which model helps generate the best summaries on average. After passing around 1000 articles of data, we found out that the Bart model outperformed the T5 model in every aspect albeit the difference was not very large. This tells us that for mid-sized news articles, the Bart Model is better than the T5 model when it comes to text-summarization.*

Key Words: **text-summarization, bart, web-scraping, t5, automation, news-summarization**

1.INTRODUCTION

With the advent of the internet, large quantities of news articles have flooded the internet. However, articles are often long and roundabout, or simply have catchy titles for increasing viewership. A summary of a news article could deliver the same message in a straightforward manner, saving time and energy. Further, news summaries can give users a quick preview of the main points of the content, giving them the opportunity to understand the gist of an article and read the whole article only if it is what they want. Previously, text summarization has also found applications in product reviews, and fiction and nonfiction books[1],[2],[3].

With the necessity of summarizing only the important information and identification of these informative points, news summarization has become a perplexing task. That being said, with the advancements taking place in Machine Learning, Artificial Intelligence, and Natural Language Processing, research on text summarization has evolved exponentially. A text summarization model that can instantly give a summary of any important topic entered as a query by the user, such as - 'Coronavirus', 'Latest News about Afghanistan', or a summary of any specific article online by passing the link to that particular article can be invaluable in today's world as you can gain a general idea about any topic within seconds. An automated text summarizer has many benefits, and by removing multiple tedious steps such as: finding an article from a trusted source. This process is tedious and time-consuming.

A complete integrated system that automates the news fetching and summarization process ensures the quick delivery of summarized articles. Furthermore, comparing different text summarization models will give us the best model to use to get the most relevant summary. You could also use the same article to generate summaries by different models and pick your favorite one. By automating the entire process, time and energy can be saved.

2. Existing Methods and Drawbacks

2.1 Manual Selection of Important points

Many websites do the whole process manually. However, this process is tedious and time consuming. It involves not only going through multiple articles from different sources and comparing them, but also manually selecting the important points from the articles. Due to the large number of manual steps, this method of news summarization takes a comparatively large amount of time and energy

2.2 Presence of irrelevant content and ads

Scraping articles online comes with the drawback of picking up on multiple advertisements and links to other unrelated articles. This useless information can end up in the summary leading to a poor summary generation. We use the newspaper3k package in python to scrape only the relevant information of the website while ignoring ads and other unimportant content.

2.3 Text summarization by humans

Conventionally, humans produce summaries of large articles, but this takes up a lot of time and money. While a model might not produce a summary as coherent and concise as a human, it can produce tens of thousands of summaries where a human would produce one. Thus, being more advantageous in the long run.

3. Our Approach

3.1 Dataset

We have made use of the BBC News Summary[4] dataset, also available on Kaggle. This dataset has news articles grouped into 5 classes – 'business', 'entertainment', 'politics', 'sport', and 'tech'. There are also summaries provided for each article in each class. This dataset cumulatively consists of more than 2200 articles with summaries.

3.2 Automated News Scraping

```
In [2]: import requests
        from bs4 import BeautifulSoup
        from newspaper import Article

In [5]: query = input("Enter search query\n")
        url = 'https://news.google.com/search?q=' + query
        response = requests.get(url)
        soup = BeautifulSoup(response.text, 'html.parser')
        a = soup.find('a', class_='DY5T1d RZIKme')
        href = a['href']
        article = Article('https://news.google.com/'+href)
        article.download()
        article.parse()
        info = article.text

        Enter search query
        bitcoin
```

Fig-1: Code for scraping news articles

We use the requests, bs4[5], and newspaper packages to automatically scrape the web and store the most relevant article without any of the ads or irrelevant links.

The requests package is used to 'GET' the desired URL page and returns us an object of that webpage. The BeautifulSoup class is used for converting the request object received and parsing it into HTML. It can then be used to scrape the desired information which in this case were the links to the most relevant article. We then store the link to that article in the variable 'href'. The Article class is used to download, parse, and store the cleaned-up article into the variable 'info' ready for summarization.

```
print(info)
Cryptocurrency (Representative image: Reuters)
Cryptocurrency prices tumbled on
 September 25 after China's central bank declared all transactions involving Bitc
oin and other virtual currencies illegal.
The global cryptocurrency market cap i
s $1.91 trillion, a 4.23 percent decrease over the last day. The total crypto mar
ket volume over the last 24 hours is $134.58B, which makes a 34.00 percent increa
se, as per Coin Market Cap.
Bitcoin's price is currently Rs 33,46,436 and its do
minance is currently 42.04 percent, a decrease of 0.08 percent over the day.
Als
o Read | Bitcoin slips after China central bank vows to crack down on crypto trad
ing
The price of Bitcoin fell over 4 percent, in the hours after China's Central
 Bank's announcement, as did most other crypto tokens.
Bank notice complained Bi
tcoin, Ethereum and other digital currencies disrupt the financial system and are
```

Fig-2: Scraped article text

3.3 BART

The Bidirectional and Auto-Regressive Transformer or BART, introduced by Lewis et al [6] is a Transformer that combines the Bidirectional Encoder (i.e. BERT like) with an Autoregressive decoder (i.e. GPT like) into one Seq2Seq model. In other words, it gets back to the original Transformer architecture proposed by Vaswani et al[7], albeit with a few changes.

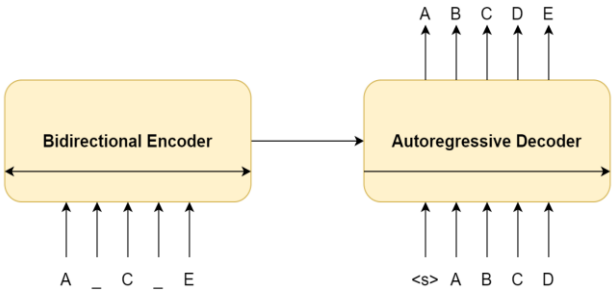

Fig-3: Bart

We make use of the pipeline function from huggingface transformers api[8] to run the Text Summarization process using the Bart Model. "device=0" implies we are using a GPU for computation. We pass a thousand records of our data from our dataset into the pipeline function to generate the summary. We make use of the rouge metric for evaluation of our model. We use the rouge-1, rouge-2, and rouge-l evaluation techniques.

```
summarizer = pipeline('summarization',device=0)

r1_fscore_f, r1_fscore_p, r1_fscore_r = [], [], []
r2_fscore_f, r2_fscore_p, r2_fscore_r = [], [], []
rl_fscore_f, rl_fscore_p, rl_fscore_r = [], [], []

for filename,article, summary in tqdm(zip(dataset['File_path'],dataset['Articles'],dataset['Summaries'])):


  bart_output = summarizer(article[:4000])
  bart_output = bart_output[0]['summary_text']


  r_bart = Rouge()
  r1_fscore_f.append(r_bart.get_scores(bart_output, summary)[0]['rouge-1']['f'])
  r1_fscore_p.append(r_bart.get_scores(bart_output, summary)[0]['rouge-1']['p'])
  r1_fscore_r.append(r_bart.get_scores(bart_output, summary)[0]['rouge-1']['r'])
  r1_fscore_f.append(r_bart.get_scores(bart_output, summary)[0]['rouge-1']['f'])
  r1_fscore_p.append(r_bart.get_scores(bart_output, summary)[0]['rouge-1']['p'])
  r1_fscore_r.append(r_bart.get_scores(bart_output, summary)[0]['rouge-1']['r'])
  r2_fscore_f.append(r_bart.get_scores(bart_output, summary)[0]['rouge-2']['f'])
  r2_fscore_p.append(r_bart.get_scores(bart_output, summary)[0]['rouge-2']['p'])
  r2_fscore_r.append(r_bart.get_scores(bart_output, summary)[0]['rouge-2']['r'])

```

Fig-4: Code for generating bart scores

3.4 T5

Raffel et al[9] present a large-scale empirical survey to determine which transfer learning techniques work best and apply these insights at scale to create a new model called the Text-To-Text Transfer Transformer (T5). They propose reframing all NLP tasks into a unified text-to-text-format where the input and output are always text strings, in contrast to BERT-style models that can only output either a class label or a span of the input.

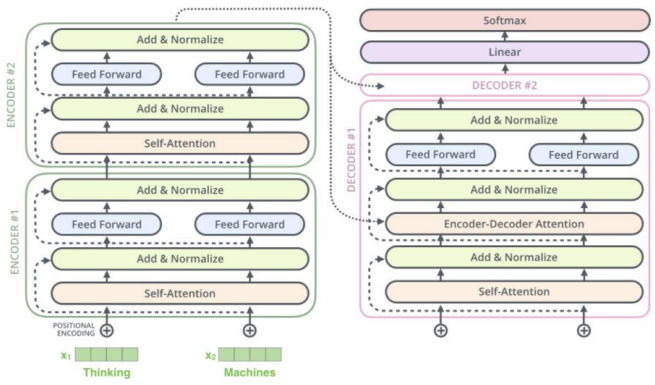

Fig-5: T5

We make use of the pipeline function from the huggingface api [8] to run the Text Summarization process using the T-5 Model. "device=0" implies we are using a GPU for computation. We pass a thousand records of our data from our dataset into the pipeline function to generate the summary. We make use of the rouge metric for evaluation of our model. We use the rouge-1, rouge-2, and rouge-l evaluation techniques.

```
summarizer = pipeline('summarization',model='t5-base',device=0)
r1_fscore_f, r1_fscore_p, r1_fscore_r = [], [], []
r2_fscore_f, r2_fscore_p, r2_fscore_r = [], [], []
rl_fscore_f, rl_fscore_p, rl_fscore_r = [], [], []

for filename,article, summary in tqdm(zip(dataset['File_path'],dataset['Articles'],dataset['Summaries'])):

  t5_output = summarizer(article[:4000])
  t5_output = t5_output[0]['summary_text']


  r_t = Rouge()
  r1_fscore_f.append(r_t.get_scores(t5_output, summary)[0]['rouge-1']['f'])
  r1_fscore_p.append(r_t.get_scores(t5_output, summary)[0]['rouge-1']['p'])
  r1_fscore_r.append(r_t.get_scores(t5_output, summary)[0]['rouge-1']['r'])
  r1_fscore_f.append(r_t.get_scores(t5_output, summary)[0]['rouge-1']['f'])
  r1_fscore_p.append(r_t.get_scores(t5_output, summary)[0]['rouge-1']['p'])
  r1_fscore_r.append(r_t.get_scores(t5_output, summary)[0]['rouge-1']['r'])
  r2_fscore_f.append(r_t.get_scores(t5_output, summary)[0]['rouge-2']['f'])
  r2_fscore_p.append(r_t.get_scores(t5_output, summary)[0]['rouge-2']['p'])
  r2_fscore_r.append(r_t.get_scores(t5_output, summary)[0]['rouge-2']['r'])

```

Fig-6: Code for generating T5 scores

4. Results

We use the rouge metric introduced by Lin[10] to evaluate results. Rouge stands for Recall-Oriented Understudy for Gisting Evaluation. It calculates the similarity between a generated summary with a list of reference summaries.

The Rouge-1 and Rouge-2 are types of Rouge-N scores where N stands for n-gram. The formula is given by

$$\text{ROUGE-N}_{\text{single}}(\text{candidate, reference}) = \frac{\sum_{r_i \in \text{reference}} \sum_{\text{n-gram} \in r_i} \text{Count}(\text{n-gram, candidate})}{\sum_{r_i \in \text{reference}} \text{numNgrams}(r_i)},$$

Fig-7: Rouge-N Formula

We also use the Rouge-L score where L stands for Longest Common Subsequence. The formula is given by

$$\text{ROUGE-L}_{\text{single}}(\text{candidate, reference}) = \frac{(1+\beta^2) R_{\text{lcs}}(\text{candidate, reference}) P_{\text{lcs}}(\text{candidate, reference})}{R_{\text{lcs}}(\text{candidate, reference}) + \beta^2 P_{\text{lcs}}(\text{candidate, reference})},$$

Fig-8: Rouge-L Formula

Each rouge score has a 'f', 'p' and 'r' metric which stands for F1-Score, precision, and recall respectively.

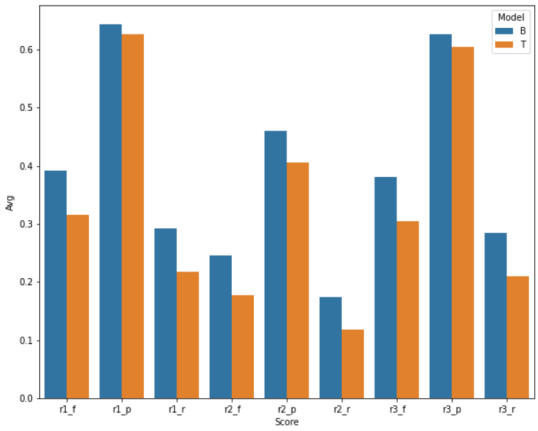

Chart-1: Score Comparison of Bart and T5

As we can see from the graph above, the Bart Model, given in blue, consistently outperforms in and generates more consistent summaries as compared to the T5 model. Given in blue.

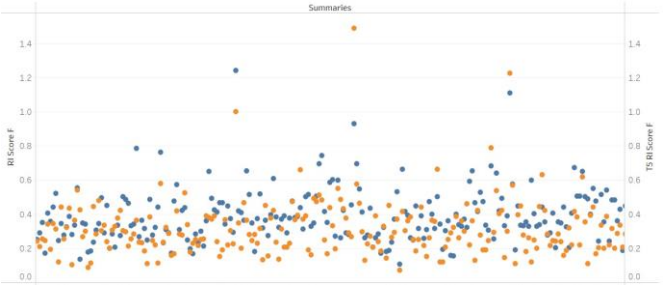

Chart-2: Rouge-1 score of every article

The graph above shows us the scatter plot of the rouge-1 F1 score of all the data points for the Bart (given in blue) and T5 Model (given in orange). The Bart model has a higher average score compared to the T5 Model.

5. CONCLUSIONS

In this paper, we compared the Bart and T5 model when it comes to text-summarization of news articles. We also prepared a method for the automatic scraping of articles from google news using just keywords related to the article. The scraping is more efficient as it picks up only relevant content while leaving the unnecessary, irrelevant content. After running the BBC dataset through our models, we found out that the Bart Model, having an average F1 score of about 33% outshines the T5 model that has an average F1 score of about 26%. It also has better metrics when it comes to recall and precision across all rouge-1, rouge-2, and rouge-l scores. Hence, we can say conclusively that the Bart model is more efficient in summarizing mid-sized news articles as compared to the T5 model.

REFERENCES

[1] I. Awasthi, K. Gupta, P. S. Bhogal, S. S. Anand and P. K. Soni, "Natural Language Processing (NLP) based Text Summarization - A Survey," 2021 6th International Conference on Inventive Computation Technologies (ICICT), 2021, pp. 1310-1317, doi: 10.1109/ICICT50816.2021.9358703.

[2] R. Boorugu and G. Ramesh, "A Survey on NLP based Text Summarization for Summarizing Product Reviews," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), 2020, pp. 352-356, doi: 10.1109/ICIRCA48905.2020.9183355.

[3] P. N. Varalakshmi K and J. S. Kallimani, "Survey on Extractive Text Summarization Methods with Multi-Document Datasets," 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2018, pp. 2113-2119, doi: 10.1109/ICACCI.2018.8554768.

[4] D. Greene and P. Cunningham. "Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering", Proc. ICML 2006

[5] Richardson, L. (2007). Beautiful soup documentation. April.

[6] Mike Lewis and Yinhan Liu and Naman Goyal and Marjan Ghazvininejad and Abdelrahman Mohamed and Omer Levy and Veselin Stoyanov and Luke Zettlemoyer (2019). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. CoRR, abs/1910.13461.

[7] Ashish Vaswani and Noam Shazeer and Niki Parmar and Jakob Uszkoreit and Llion Jones and Aidan N. Gomez and Lukasz Kaiser and Illia Polosukhin (2017). Attention Is All You Need. CoRR, abs/1706.03762.

[8] Thomas Wolf and Lysandre Debut and Victor Sanh and Julien Chaumond and Clement Delangue and Anthony Moi and Pierric Cistac and Tim Rault and Rémi Louf and Morgan Funtowicz and Jamie Brew (2019). HuggingFace's Transformers: State-of-the-art Natural Language Processing. CoRR, abs/1910.03771.

[9] Colin Raffel and Noam Shazeer and Adam Roberts and Katherine Lee and Sharan Narang and Michael Matena and Yanqi Zhou and Wei Li and Peter J. Liu (2019). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. CoRR, abs/1910.10683.

[10] Lin, C. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. ACL 2004.