Exploring Summarization Techniques for Scientific Papers

Arnab Arup Gupta, **Jaimin Dineshbhai Khanderia**, **Jaya Mundra**, **Nidhi Ranjan**
New York University
{ag7654, jk7178, jm8834, nr2387}@nyu.edu

Abstract

Automatic text summarization has offered many approaches to solving the overwhelming influx of research papers, categorized into extractive and abstractive methods. Our work provides a comparative analysis of two models in each category, scored using ROUGE and BLEU metrics in text summarization. We explore TextRank, BERT, XLNet and Pointer-Generator models.

1 Introduction

Text summarization means generating a concise and precise summary of voluminous texts while focusing on the sections that convey useful information without losing the overall meaning. Automatic text summarization aims to transform lengthy documents into shortened versions, which would be difficult and costly to undertake manually. This has become an important problem in research, with the requirement for an automatic summarization tool at an all-time high.

As scientific research papers are of much importance in the current research fraternity, we aim to apply the SOTA text summarization techniques (extractive as well as abstractive) for scientific papers. We then perform a comparative analysis of the results obtained from all the techniques on our dataset using the BLEU and ROUGE metric. We observe that in general the extractive text summarization techniques perform better than the abstractive text summarization techniques as the metric tries to check for the words that are present both in the gold summary and the generated summary, more ideally used to score verbatim reproduction.

2 Related Work

Recent research in summarization techniques has focused on neural methods to generate summaries, which can be very data-hungry. Earlier works on summarization of scientific publications were done on very small datasets, consisting of tens of documents [6]. Training on such small datasets was not sufficient for supervised summarization models relying on neural methods for sentence and document encoding to generate proper summaries because they need to be trained on many thousands of documents. More recent work on summarization of research papers was done on a larger dataset containing 10k documents [1].

Both extractive and abstractive summarization techniques were used to train the model and generate summaries. Early work on extractive summarization used statistics like word frequency to generate summaries[8]. Recent methods in extractive summarization focus on neural approaches, based on bag of word embeddings or encoding whole documents[5].

The first application of modern neural network for abstractive text [11] summarization used attention mechanism and recurrent decoders on two sentence-level summarization datasets. However, research on datasets with longer texts was infrequent and the initial work[3] was done adapting the DeepMind question-answering dataset, resulting in the CNN/Daily Mail dataset which provided the first abstractive baselines. Further research on hybrid approach using Pointer generator models[14] which is a sequence-to-sequence model that uses attention and coverage, produces an output sequence consisting of elements from the input sequence.

For our work, we experiment we both extractive and abstractive approaches to compare both the techniques and models for performance and accuracy. We use TextRank and BERT to develop extractive summaries and XLNet and Pointer generative model using coverage to generate abstractive summary and analyze the result.

3 Methods

3.1 Problem Statement

The volume of scientific literature published each year is overwhelming for students and researchers, and one needs to get through all this information quickly. At the same time, it is vital to not miss out on any critical information. The solution to this dilemma of trading time for information coverage is the advent of Text Summarization.

Implementing automatic summarization can enhance the readability of documents and reduce the time spent researching for information. While abstractive methods have a higher chance of capturing more information working within a word-limit, in the case of research papers, it is important to sometimes capture certain information verbatim. There are well-established SOTA models for both kinds of text summarization that we will apply to datasets of scientific papers and conduct a comparative analysis of the results from those experiments.

3.2 Approach

In our work, we have performed a comparative analysis of extractive and abstractive approaches to Text summarization.

Extraction-based summarization approaches generate a summary by selecting important existing words, phrases, or sentences from the original text. The words or phrases are rearranged slightly to give the sentence a structure. This method is analogous to using a highlighter to highlight important sentences in a paper, as most of us often do in a manual setting. This method maintains the information from the original paper verbatim, introducing no unseen words or phrases.

This is the starting point of our study, and we will work with the **TextRank** [9] algorithm and **BERT** [2] models in this category.

The extraction-based summarization algorithm firstly generates an importance score for each sentence. The TextRank algorithm is a graph-based unsupervised text summarization algorithm, where sentences are modeled as vertices and the ranking of vertices follows the same algorithm as the PageRank algorithm. The highest-ranked vertices are then selected to form the summary.

The motivation behind studying TextRank is that it is a popular unsupervised algorithm. It is lightweight, doesn't require a large corpus, is language independent and is a well-developed and well-researched algorithm, often used as a benchmark for research in text summarization.

Alongside this, we use BERT sentence embeddings to build an extractive summarizer taking supervised approaches which would incorporate sequential information. BERT (Bidirectional transformer) is a pre-trained bidirectional transformer model that jointly conditions on both left and right contexts, with attention mechanism. It is used to overcome the limitations of RNN and other neural networks as Long term dependencies are well handled without increasing the computation cost.

Following this, we focus on the **Abstraction-based summarization** approach, which generates a summary by taking out the important word, phrase, or sentence from an original text document and rephrasing it with proper synonyms. This method is analogous to reading the paper and writing in your own words a summary combining the most important points. It tends to be more complicated than the extraction-based technique due to the involvement of figuring out the correct synonyms and correctly rephrasing the sentence. In this category, we train and analyse the **XLNet** [15] model and **Pointer-generator Models** [12].

We fine tune XLNet to do abstractive summarization over an arbitrarily long corpus. XLNet is pre-trained recurrent language model which is widely used for language generation. They were designed to model extremely long sequences by breaking those sequences into chunks and processing them one at a time and memorizing recursively between the forward pass to the next. It overcomes the shortcomings of BERT due to its autoregressive nature, based on the Transformer-XL architecture.

The pointer-generator model augments the standard sequence-to-sequence because it uses a hybrid pointer-generator network that can copy words from the source text via pointing, which aids accurate reproduction of information, while retaining the ability to produce novel words through the generator; and it uses coverage to keep track of what has been summarized, which discourages repetition. We use it to overcome the two shortcomings of neural sequence to sequence models: they are liable to reproduce factual details inaccurately, and they tend to repeat themselves. Since this model is a hybrid of extractive and abstractive methods, it is expected to overcome the shortcomings of both.

Finally, we make use of ROUGE and BLEU

scores to compare the aforementioned four models.

4 Experiments

4.1 Setup

This section talks about how we collected and parsed the dataset of scientific papers so that we could convert them into a representation that could be fed into the SOTA models for text summarization. The following points explain the different phases of the dataset preparation task:

Dataset collection: We downloaded the dataset of scientific research papers from ScienceDirect which contains a lot of publications. We had to generate an API key using the NYU institutional email address on Elsevier and connect to the NYU VPN to make requests to the API to download the full content of the scientific paper. We downloaded around $\sim 10K$ scientific papers for our experiments which were in XML format and converted to a text format which is mentioned in the next section.

Dataset pre-processing: All the downloaded papers followed a general template of these sections: Title, Abstract, Highlights, Keywords, Introduction, Methodology, Experiments, Results and Conclusion. The highlights contained sentences written by the paper authors to give a brief idea about their paper and it didn't contain any sentences already present in the paper. As the highlights contains the main takeaway of the paper and as they also don't depend on the contents of the paper, we choose that to be the gold summary of the paper.

As the downloaded papers were in XML format, we had to first convert them into a useful format first and so we preprocessed the papers to convert them to a text based format. Then, the papers were parsed gather the heading and the content for those headings which were saved to a text file that was later on used for a representation to be fed into the models.

Additionally, all the experiments were carried out on the NYU Greene cluster to better utilize the computational resources needed to run the experiments.

4.2 Results

The results from our experiments, in terms of the two metrics that we used for our work, has been presented in Table 1. We compute three kinds of Rouge scores: Rouge-1, which accounts for unigram overlap; Rouge-2, which accounts for bigram overlap; and, Rouge-L, which detects the Longest Common Subsequence in both the reference and the candidate texts. The BLEU score is a weighted score of unigram, bigram, trigram and quadgram overlaps. Moreover, BLEU scores penalize for the brevity of output. So, in general, these scores have a lower value for all our models as compared with ROUGE scores.

Model	Rouge-1	Rouge-2	Rouge-L	BLEU
TextRank	0.447	0.245	0.257	0.206
BERT	**0.510**	0.277	0.305	**0.268**
XLNet	0.443	0.236	0.241	0.168
PGN*	0.502	**0.284**	**0.313**	0.252

Table 1: Accuracy scores.

*Pointer-Generator networks

As is evident from the Graph 2, the BERT model outperforms TextRank in the extractive category, whereas the Pointer-Generator model outperforms XLNET in the abstractive category. Overall,the Pointer-Generator model performs slightly better than the BERT model considering the ROUGE-L [7] score, but falls short of the same even more slightly for other rouge metrics. Their BLEU scores are also very slightly disparate with BERT[10] outperforming PGN model.

4.3 Analysis

There is a considerable difference between the Rouge scores of each model, with Rouge-1 scores almost twice that of Rouge-2 and a lot higher than Rouge-L scores. The reason behind this is that Rouge-1 scores simply measure the unigram overlap, the number of common unigrams between the reference and generated summaries. While a good Rouge-1 score is important for a decent summary, it does not always ensure the most semantically accurate collection of sentences. In this regard, we are better off referring the Rouge-L scores since they are based on the longest common subsequence between the reference and generated summaries, and are therefore a good measure of the semantic closeness of two summaries.

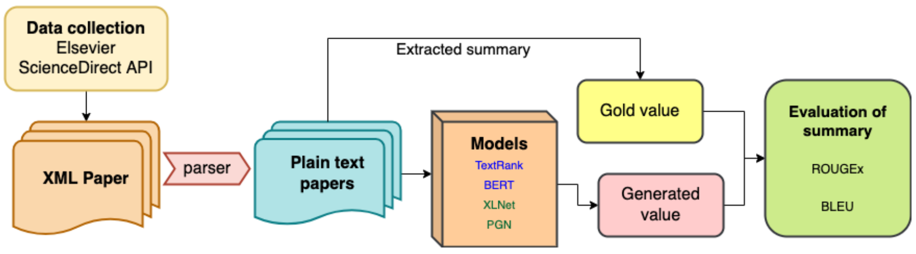

Figure 1: This figure depicts our workflow in this project.

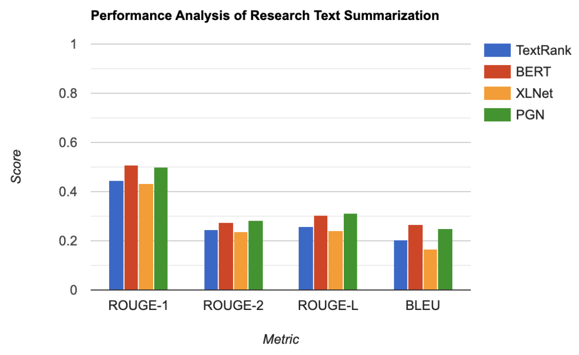

Figure 2: Evaluation of Text Summarization techniques on Research Papers

The BLEU score has been augmented with a Smoothing function which is advisable in the case of smaller summaries, as is our use case. BLEU scores are generally lower for all our models, since they are generated from a combination of unigram, bigram, trigram and quadgram overlaps, and we have given more importance to unigram and bigrams than others, since our summaries are shorter in length. Moreover, BLEU scores penalize on the brevity of the output text.

From our experiments, we can assert with some certainty that extractive models outperform abstractive models in the summarization of research papers. This is fairly intuitive since research papers comprise of domain-specific vocabulary which can often not be paraphrased or synonymized.

The four scores we have considered rank these models differently. While BERT has the best Rouge-1 score, it falls short of the Pointer Generator model (PGN) when it comes to Rouge-2 or Rouge-L. The PGN's accuracy is attributed to its hybrid nature which accounts for exact overlaps as well as some semantically sound aggregation. Words are generative to an extent, but their meaning is maintained by the extractive aspect of the model.

We should take these scores with a grain of salt considering the nature of our data set. A good score indicates a good summary, but a bad score might does not necessarily mean our summary is bad.

5 Conclusion

In this paper, we have downloaded the dataset from ScienceDirect through their API to collect the scientific research papers. We have then performed a comparative analysis of the SOTA text summarization techniques for those scientific research papers using the ROUGE and BLEU evaluation metrics. We observe that in general for our use case, the extractive text summarization techniques perform better in terms of score than the abstractive text summarization techniques considering the ROUGE and BLEU evaluation metrics. Pointer Generative Networks perform the best of all the techniques that were considered for our experiments as it is a hybrid model of extractive and abstractive models which overcomes the shortcomings of abstractive techniques by generating new words and also checking for repetition of already generated words.

Although the results from our experiment are as expected, we can improve them with some other approaches in future. We can work on the data preprocessing of the scientific papers to better handle the equations or tables present in the paper. Further, we can consider to use paragraph embedding for better representation of the document which can then be fed into the models for better results. Moreover, we can try to gather a large dataset of scientific research papers and train the models for a longer period of time to improve the results. Lastly, we should devise a new metric that compares the semantics between the gold and generated summary as compared to overlap of words that occur in both gold and generated summary.

References

[1] E. Collins, I. Augenstein, and S. Riedel. A supervised approach to extractive summarisation of scientific papers. *CoRR*, abs/1706.03946, 2017.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. pages 4171–4186, June 2019.

[3] C. Gulcehre, S. Ahn, R. Nallapati, B. Zhou, and Y. Bengio. Pointing the unknown words. pages 140–149, Aug. 2016.

[4] V. Gupta, P. Bharti, P. Nokhiz, and H. Karnick. SumPubMed: Summarization dataset of PubMed scientific articles. pages 292–303, Aug. 2021.

[5] H. Kobayashi, M. Noguchi, and T. Yatsuka. Summarization based on embedding distributions. pages 1984–1989, Sept. 2015.

[6] J. Kupiec, J. Pedersen, and F. Chen. A trainable document summarizer. page 68–73, 1995.

[7] C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. pages 74–81, July 2004.

[8] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165, 1958.

[9] R. Mihalcea and P. Tarau. TextRank: Bringing order into text. pages 404–411, July 2004.

[10] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. pages 311–318, July 2002.

[11] A. M. Rush, S. Chopra, and J. Weston. A neural attention model for abstractive sentence summarization, 2015.

[12] A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. *CoRR*, abs/1704.04368, 2017.

[13] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

[14] O. Vinyals, M. Fortunato, and N. Jaitly. Pointer networks. 2017.

[15] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. 2020.