Extractive Summarization - A Comparison of Pre-Trained Language Models and Proposing a Hybrid Approach

Jigisha M Narrain
Computer Science and Engineering
PES University
Bangalore, India
jigishanarrain@yahoo.in

Vanshika Taneja
Computer Science and Engineering
PES University
Bangalore, India
vanshikataneja125@gmail.com

Sanjana B Atrey
Computer Science and Engineering
PES University
Bangalore, India
sanjanaatrey@gmail.com

Jahnavi Sivaram
Computer Science and Engineering
PES University
Bangalore, India
jahnavisivaram123@gmail.com

Dinesh Singh
Computer Science and Engineering
PES University
Bangalore, India
dineshs@pes.edu

2023 Winter Summit on Smart Computing and Networks (WiSSCoN) | 979-8-3503-3405-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/WISSCON56857.2023.10133863

Abstract*—The automatic summarization of technical articles is a field that has garnered a fair amount of interest, and one that enjoys a significant portion of NLP-related research. As a whole, automatic summarization can be split into two broad categories - extractive and abstractive. Extractive summarization implies that important and relevant sentences are picked from the article as is, and inserted in the summary. Abstractive summarization, on the other hand, requires contextual understanding of the document, and rearranging and shortening the sentences, while maintaining the core essence of the article. Multiple algorithms have been proposed for both these classes of automatic summarization. In the recent past, the emergence of pre-trained language models for NLP tasks have been heralded by the creation of attention mechanisms and Transformers. These models implement encoder-decoder structures, and have far wider applications than previously utilised algorithms. Four such pretrained models are - BERT, BART, XLNet and GPT-2. In this project, we use transfer learning to fit these models on our corpus of Medium articles, fine-tuning them for the task of summarization. Further, we generate online summaries of the data, and use ROUGE metrics to evaluate the accuracy of the model-generated summaries alongside those. We also implement the popular Word2Vec model on the same data, and compare its result with the ones obtained from the attention based models, once again using ROUGE metrics. In addition, we explore the effectiveness of a hybrid approach to the summarization task, by using different combinations of the models on the same article to investigate the results of the same.

***Index Terms*—natural language processing, machine learning, automatic summarization, language model, attention mechanism, BERT, BART, XLNet, GPT-2, hybrid model, word2vec**

I. Introduction

The act of expressing the most important facts or ideas about something or someone in a short and clear form is referred to as summarization [1]. Automatic summarization is carrying out this process computationally. It broadly falls into two categories - extractive summarization and abstractive summarization.

Extractive summarization techniques take the already existing phrases from the text and present them in the summary, choosing relevant parts. This method does not create new words or phrases.

Abstractive summarization creates new words or phrases based on the content of the text in a way that most accurately and concisely represents the original subject matter. It logically follows that abstractive summarization methods are more complex and computationally intensive than extractive ones.

A recent innovation in technologies that enable summarization tasks is the emergence of pre-trained language models. These are a subset of natural language processing in machine learning that have garnered significant interest and traction since the development of one such pre-trained language model, BERT, in 2018 by researchers at Google AI Language [3].

This paper does a comparative study of the performance of four such language models that have been developed over the last five years - BERT, BART, XLNet, and GPT-2 - in the task of extractive summarization. All models are run over textual data, and the resulting summary is evaluated against online generated summaries [2] using ROUGE metrics. In an effort to improve performance, the individual models are applied in different orders and combinations on the same text to implement a hybrid approach. In addition, a Word2Vec procedure is also followed to substantiate the claim that it could lead to lossy results when compared to the state of the art ones obtained through more modern, machine learning-based tools. The dataset being used has been obtained from Kaggle [24], and comprises articles from the popular knowledge-sharing website, Medium [11].

The paper is divided into nine sections. Section 2 talks about the history of automatic summarization, and significant breakthroughs in the past related to the field. Section 3 logically follows as a literature survey, consisting of summaries of relevant and important past works that have been integral for our research. Section 4 is a deep dive into the four models of focus, with information regarding their development, use cases, and specific differences. It also includes the background and usage of the Word2Vec algorithm. The dataset used and its characteristics have been elaborated on in section 5. In section 6, the usage and specifics of the ROUGE metrics have been explained, along with further clarification of the particular flavours of ROUGE employed on this research. Section 7 highlights the method of implementation of the models and the Word2Vec algorithm against the text. Tabulated observations and relevant insights are provided in section 8. In this section, the outputs of ROUGE metrics and the efficiency of hybrid models have been explored using various relevant tables and graphs. In section 9, the scope for further study and potential future and newly developed techniques for extractive automatic summarization have been expounded. It also concludes the paper.

II. History of Summarization

The task of automatic summarization was initially explored by Luhn in 1958. In 'The Automatic Creation of Literature Abstracts' [4], he carries out an exploratory study, and derives textual summaries by gauging the statistical importance of terms through their frequency and distribution. Thus, a relative measure of significance is obtained, and sentences with the highest significance score are added to the result summary.

Since then, the field of computer-generated summarization has undergone several iterations of innovation. Methods such as Word2Vec, frequency-based, TextRank, and latent semantic analysis are all techniques that have been developed for the task of document summarization. Other approaches that have been developed with varying degrees of success are fuzzy logic-based methods, linguistic methods, and concept methods [5]. A widely popular subset of this research revolves around multi-document summarization, which aims at extracting information from multiple texts that have been written on the same topic.

With the rapid and large-scale changes being made to the computer science research landscape, the field of machine learning has led to further exploration into the field of automatic summarization. Natural language processing has lent new methods and techniques for automating core text-based tasks like annotation, keyword extraction, and part-of-speech tagging. Within the field of automatic summarization, by far the most impactful development has been that of language models.

A comparatively recent, and constantly in-progress advancement, language models, usually pre-trained by large teams with a wide variety of resources, provide state of the art results. Leveraging the precision and training capability of these systems has led to more accurate and concise summaries, the quality of which non-machine learning techniques could not reach.

Alongside the continuous progress in automatic summary creation, the task of automatic summary evaluation has also undergone several modifications over the years, each change better reflecting the needs of users and the particular advancement of the time. The most popular is the ROUGE metric, which is utilised in this paper and further explained in section VI. Variations of it include ROUGE-WE [6] and S3 [7]. Recent developments include SummEval [8], BLANC [9], and BERTScore [10]. All of these evaluation techniques have different efficiencies, use cases, and baseline approaches.

The current state of automatic summarization lies at iterations to machine learning models, with research related to parameter tuning and changes that can be made to obtain the best possible summary from existing methods. CNN-based deep learning methods, graph-based approaches, and the combination of extractive and abstractive techniques are also becoming popular fields of research.

III. Literature Survey

In their paper titled 'SummEval: Re-evaluating Summarization Evaluation' [8], Fabbri, Kryściński et al. focus on the need for a common evaluation protocol of summaries along five dimensions, using different neural network models and the CNN/DailyMail dataset. ROUGE, an automatic evaluation metric has, despite its shortcomings, become a standard, which they say could be inhibiting progress in this field. The different evaluation metrics that have been compared in their paper include ROUGE, BLEU, BertScore, and Data Statistics.

The extractive summarization techniques that are explored here include NEUSUM, BanditSum, and STRASS.

By evaluating each model's outputs using the factors of coherence, consistency, fluency, and relevance, the authors compare how the methods and metrics match up with each other, and present SummEval – '. . . a set of resources for summarization model and evaluation research'. This in-depth analysis gives us a cohesive stockpile to consider for evaluation of the output procured by the model that we have developed, allowing us to explore accuracy and efficiency of our project along different parameters highlighted here.

Liu and Lapata [14] introduce a document-level encoder that is based on Bidirectional Encoder Representations from Transformers (BERT) to express the semantics of a document in automatic summaries. The paper explores the role that language-model pre-training plays on text summarization. It also shows that to generate better summaries, a two-step process, which includes both extractive and abstractive approaches, is useful.

The authors experiment with three different datasets using this method, and show that it ensures sophisticated results under both automatic metrics and human-based protocols. Their research introduces us to BERT, as well as encoding as a method for sentence representations, which enables further investigation into the natural language processing techniques we use for pre-processing our dataset, as well as optimising the results obtained.

In their paper titled ’Deep reinforcement and transfer learning for abstractive text summarization: A review’ [15], the authors outline a thorough review of recent ATS advances. Problems faced are described, as well as their proposed solutions. Furthermore, abstractive ATS datasets and evaluation metrics are discussed. Since summary correctness is hard to gauge, ROUGE is considered to be the standard evaluation metric for the same. Recall, precision, and F1 score are usually utilised. The amount that the generated summary captures the reference summary is measured by recall. Precision overcomes the preceding issue by determining how much of the resulting summary is relevant. By determining their harmonic mean, F1 balances both recall and precision grades. ROUGE-F1 is the most popular standard used to measure three ROUGE score varieties - ROUGE-1 (unigram overlap), ROUGE-2 (bigram overlap), and ROUGE-L (longest common subsequence). However, the evaluation phase’s ROUGE is incompatible with the training phase’s cross-entropy loss function. In addition, it is not differentiable. Reinforcement learning adeptly handled these limitations. Meteor and BLUE are other ATS evaluation metrics that are utilised.

In Attention Is All You Need [16], it is learnt that sequence-to-sequence models in deep learning consist of an encoder and a decoder. The encoder concerts its input into a vector, and passes this vector as the input to the decoder, which outputs another sequence of text. The Attention technique ensures that models can focus on the relevant aspects of the text sequence. The Transformer is a Seq2Seq model that is based solely on attention mechanisms. Unlike previous research in this domain that combines attention methods along with recurrent networks, which employ hidden states, this paper proposes a model that does not include recurrent networks. It replaces the commonly used recurrent layers with multi-headed self-attention, thus improving the performance of the attention layer. Multi-headed self-attention means the model can focus on different parts of the sentences, even while it is currently concerned with a single words’ encoding. It also gives the attention layer multiple representation subspaces that are randomly initialised. Post-training, per Seq2Seq modelling objectives, each set is used to project the input embedding into a different representation subspace.

Transformer-XL [17] is a neural architecture that builds on the existing Transformer architecture, and removes the latter’s limitation of fixed-length context. It also resolves the Transformer’s context fragmentation issue, where it does not respect sentence boundaries. RNN-employing techniques for language modelling, such as Long Short Term Modelling (LSTM), are widely effective, but difficult to optimise due to explosion and the vanishing gradient problem. Further, other models that work on fixed context length inputs cannot be implemented on longer term dependencies. Transformer-XL is a neural architecture that builds on the existing Transformer architecture, and removes the latter’s limitation of fixed-length context. It also resolves the Transformer’s context fragmentation issue, where it doesn’t respect sentence boundaries. RNN-employing techniques for language modelling, such as Long Short Term Modelling (LSTM), are widely effective, but difficult to optimise due to explosion and the vanishing gradient problem. Further, other models that work on fixed context length inputs cannot be implemented on longer term dependencies. Transformer-XL introduces recurrence into the network, by reusing the hidden states from previous segments, acting as memory and building recurrence between segments. This enables the modelling of long-term dependency. Relative positional encodings are used instead of absolute ones, to avoid temporal confusion.

The introduction of XLNet by Yang at al. [18] deals with the drawback of BERT, which is that it relies on input corruption using masks. This leads to pretrain-finetune discrepancy. The authors present XLNet as an alternative, autoregressive pre-training model, which overcomes this limitation. Previous experiments in unsupervised pre-training have shown that autoregressive language modelling and autoencoding are successful objectives. XLNet includes Transformer-XL, which combines the Transformer’s attention modules and a recurrence mechanism on each consecutive input segment, thus improving the model’s performance on longer text. To remove ambiguity in the usage of Transformer-XL, they also reparameterize it. In addition, it is found that BERT’s next-sentence prediction objective does not lead to an improvement in results, and is hence not included in XLNet. This procedure combines the advantages of AR and AE methods, while keeping in mind the objectives.

The BART model is presented in the paper titled ’BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension’ [19]. An acronym for Bidirectional Autoregressive Transformers, it is a denoising autoencoder, which incorporates the mechanism of bidirectional encoding from the BERT model and the left to right decoding from the GPT model. Using the SequenceToSequence and Transformer architecture, it learns reconstruction of corrupted text. The model is trained on corrupted, noisy text where phrases are replaced by a single masking token, called text infilling, or document tokens are simply deleted. Evaluation is done by experimenting with different noisy data, shuffling sentences and varying the length of phrases to be masked. This model is capable of performing a wide range of tasks such as summarization, abstractive dialogue and answering questions. The model in this paper implements 6 layers of encoders and decoders. The encoder has a final hidden layer which is fed to each decoder layer. These layers perform a cross-attention mechanism on the input layer. The main goal of the authors' model is reconstruction of any corrupted text with optimal reconstruction loss. The model is able to compute various noise schemes to produce a 6 ROUGE score. It performs accurate extractive summarization on the CNN/DailyMail dataset.

In 'Language Models are Unsupervised Muktitask Learners' [20], GPT-2, a 1.5B parameter Transformer-base language model, achieves SOTA results in numerous language modelling tasks. Previous machine learning systems, while effective when used with their intended data, lack in accuracy when they are implemented on different distributions. However, GPT-2 is designed with a generic goal, one that can perform many tasks, without manual creation and labelling. Insofar, machine learning models are dependent on supervised training for task performance. Research has shown that unsupervised or minimal training can lead to improvement and greater generalisation of language models. This paper connects the two approaches, and shows that zero-shot learning in language modelling can be used to obtain task-based SOTA results. It also says that with a large enough language model and a diverse dataset, good performance can be obtained across multiple domains.

IV. Models Implemented

The following models have been implemented in this paper for the task of extractive summarization.

A. BERT

An acronym for Bidirectional Encoder Representations, BERT learns information from both the left and right sides of a token's context. Its pre-training consisted of 2 NLP tasks -

1. **Masked Language Modeling** - from the first layer to the last layer, Masked Language Modeling captures information from both the right and left context of a token.

2. **Next Sentence Prediction** - For tasks that demand a comprehension of the link between sentences, BERT is also trained on Next Sentence Prediction.

BERT can be fine-tuned to construct state-of-the-art models for a range of NLP tasks by just adding a few additional output layers.

B. BART

BART has a sequence-to-sequence/machine translation architecture that includes a bidirectional encoder (similar to BERT) and a left-to-right decoder (like GPT). It is a denoising autoencoder for pre-training sequence-to-sequence models, and is trained by corrupting text with a noising function, and reconstructing the original text by learning a model.

The pre-training task entails changing the order of the source phrases at random and using a novel in-filling strategy that replaces text spans with a single mask token. It is built on the Transformer neural machine translation architecture. BART is notably useful for text generation, but it also does well in comprehension tasks.

C. XLNet

XLNet uses regular language modelling architecture. Its main contribution to NLP tasks is its training objective which is 'learns conditional distributions for all permutations of tokens in a sequence' according to Borealis AI [21]. It predicts each word in a sequence using any combination of other words in the sequence, and gets an ambiguous input, which means it needs to look at all possible permutations to predict the output sequence. In autoregressive models, a word is predicted based on some combination of tokens around it. XLNet uses all such permutations.

D. GPT-2

Developed by OpenAI, GPT-2 has been trained on a massive dataset called WebText, scraped from the Internet by OpenAI scientists. It was initially trained for word prediction, and implements a decoder-only architecture. It predicts one token at a time, and adds the newly predicted token to the input sequence for the next token. In self-attention, like in BERT, the model can see both sides of the token. However, in masked self-attention, as in GPT-2, the model can only see previous tokens. GPT-2 also uses Byte Pair Encoding, where each token is considered to be a part of a word.

E. Word2Vec

Using Word2vec, word embedding can make natural language computer-readable. Word similarities can be identified by performing additional mathematical operations on them. For example, the words "women," "men," and "human" might all be in the same corner, while "yellow," "red," and "blue" might all be in the same corner. While working with the document, Word2Vec generates the summary and extracts key sentences using the basic text rank algorithm. A common graph-based summarization algorithm based on PageRank, TextRank. It creates a graph out of a document, with each vertex representing a sentence. An edge indicates that the content of the two sentences that correspond to linked vertices is the same. The number of lexical overlaps divided by the sum of two sentence lengths is the edge weight, which indicates the strength of the connection between the sentence pair.

V. Dataset

The dataset used in this paper has been obtained from Kaggle, and can be found here [24]. It consists of a collection of articles about machine learning, artificial intelligence, and data science, obtained from the Medium website [11]. The specifications of the dataset are as follows -

- 337 records
- 6 columns -
    1) Name of author
    2) Number of claps received by the article
    3) Reading time
    4) Article link
    5) Title of the article
    6) Article text
- CSV format

Relevant to this project is the final column, article text, which is the textual matter that is summarized.

About Medium—Medium is an online publishing platform that was launched in 2012. Widely used to share information and knowledge on a large gamut of topics, from both scientific and artistic fields, it is also a veritable repository of articles related to computer science and technological innovation. As students of computer science, Medium has been a useful website for keeping up-to-date with the latest cutting edge technologies, as well as reinforcing the implementation of concepts taught. Having summarized versions of these articles would be incredibly advantageous and practical.

VI. ROUGE Metrics

ROUGE, short for Recall-Oriented Understudy for Gisting Evaluation was first proposed by Lin in ROUGE: A Package for Automatic Evaluation of Summaries [12]. The paper highlights four different flavours of the metric - ROUGE-N, ROUGE-L, ROUGE-W, and ROUGE-S. The focus in the implementation followed in this paper is on the first two varieties.

A. ROUGE-N

ROUGE-N compares a candidate summary with a set of one or more provided reference summaries. Here, the 'N' refers to an n-gram. Simply put, an n-gram is nothing but a continuous sequence of words in a document. 'N' can take any positive integer, and is classified based on its value into unigram, bigram, trigram, and so on.

The ROUGE-N measure is a recall-based one, that favours candidate and reference summaries with a larger number of common n-grams. In essence, it is the ratio of n-grams found in the candidate and reference summaries to the n-grams found in the reference summary.

B. ROUGE-L

The 'L' in ROUGE-L stands for Longest Common Subsequence (LCS). Lin provides two distinct flavours of ROUGE-L, both of which can be implemented in a convenient manner in the environment utilised in this project[13]. As the term suggests, a common subsequence is a subsequence that's common to two strings. ROUGE-L is a measure that involves providing results that focus on a baseline intuitive concept - longer the LCS between the reference and candidate summary, greater the score provided.

The two specifications of ROUGE-L are sentence-based LCS and summary-based LCS. The difference between the two lies in the way each of them deals with newlines.

Sentence-based LCS ignores newline characters. However, summary-based LCS actually interprets newlines as sentence boundaries. After each set of candidate sentence and reference sentence is compared, the final ROUGE score is provided based on a union-summary that is computed.

VII. Implementation

The text is first cleaned by removing the hyperlinks, escape characters, and words in square brackets. The four models have been obtained from their respective HuggingFace Pytorch packages [22]. For Word2Vec, the spacy library, along with the heapq module, has been utilised. Implementation of the ROUGE metric has been done using Python's native package, rouge-score [13]. The entire code can be found here.

BERT - The summary length was fine-tuned using the ratio parameter that specifies the percentage of the original article to be included in the summary.

BART - The method of tokenization, encoding, and decoding was done to summarize large texts. The function to fit the pre-trained model, summarizer(), could not accommodate a range of tokens greater than 1024, and was hence not used.

GPT-2 - The transformer-type used is GPT-2, with the min_length and max_length parameters being repeatedly modified to obtain different results.

XLNet - The transformer type used is xlnet-base-cased, which considers both upper and lower cases, and is the smaller version of the large case. The parameters considered for fine-tuning the model were min_length and max_length.

Hybrid Models - Models are put into pairs, and each pair is implemented on the same text, such that one half of the text is run through one model and the other is run through the other model. On combining the two half-summaries, the entire summary for the full text is obtained, and evaluated as a single block of text.

Word2Vec - The word frequencies are first calculated and normalised. Subsequently, sentence tokens are computed, and the sentences are scored based on the previously calculated word frequencies. Using the nlargest() method from the heapq module, a summary is generated after providing the required output length, and sentence scores obtained.

ROUGE - ROUGE-1 (unigrams) and ROUGE-L have been implemented, using the RougeScorer() and score() methods.

VIII. Results and Discussion

The resultant summaries were evaluated using two methods - ROUGE metrics and an individual article-model comparison. The Word2Vec results have been presented separately.

A. ROUGE Evaluation

Three texts, namely Text-0 (1981 words), Text-7 (995 words), and Text-25 (1941 words), have been put through all four models. The resultant summary was evaluated against a sample summary using two flavours of ROUGE - Rouge-1 and ROUGE-L. The parameters considered are precision (**TP/(TP+FP)**; *proportion of words suggested by the candidate summary, that actually appear in the reference summary*) and recall (**TP/(TP+FN)**; *proportion of words in the reference summary captured by the candidate summary*)[23]. The results obtained have been tabulated as follows -

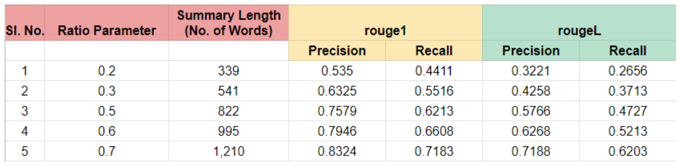
TABLE I
ROUGE-1 for Text-0

Parameter	BERT	BART	GPT-2	XLNet
Precision	0.6455	0.6138	0.5815	0.3965
Recall	0.7703	0.3508	0.4121	0.3876

TABLE II
ROUGE-L for Text-0

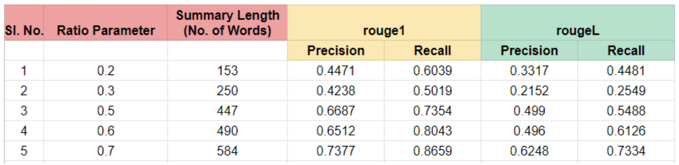
Parameter	BERT	BART	GPT-2	XLNet
Precision	0.4988	0.2023	0.4449	0.1552
Recall	0.5952	0.1156	0.3070	0.1517

After obtaining these initial results, each of the models were fine-tuned according to their respective parameters, and repeatedly evaluated. All optimal parameter values were deduced keeping in mind the generally preferred length of a summary, which is 30% of the original text. The subsequent observations were -



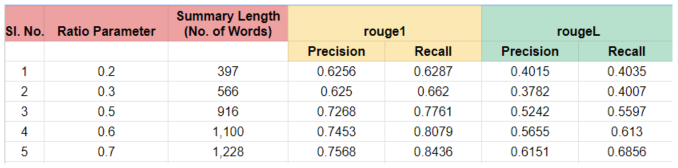
Sl. No.	Ratio Parameter	Summary Length (No. of Words)	rouge1		rougeL	
			Precision	Recall	Precision	Recall
1	0.2	339	0.535	0.4411	0.3221	0.2656
2	0.3	541	0.6325	0.5516	0.4258	0.3713
3	0.5	822	0.7579	0.6213	0.5766	0.4727
4	0.6	995	0.7946	0.6608	0.6268	0.5213
5	0.7	1,210	0.8324	0.7183	0.7188	0.6203

Fig. 1. BERT scores for Text-0



Sl. No.	Ratio Parameter	Summary Length (No. of Words)	rouge1		rougeL	
			Precision	Recall	Precision	Recall
1	0.2	153	0.4471	0.6039	0.3317	0.4481
2	0.3	250	0.4238	0.5019	0.2152	0.2549
3	0.5	447	0.6687	0.7354	0.499	0.5488
4	0.6	490	0.6512	0.8043	0.496	0.6126
5	0.7	584	0.7377	0.8659	0.6248	0.7334

Fig. 2. BERT scores for Text-7

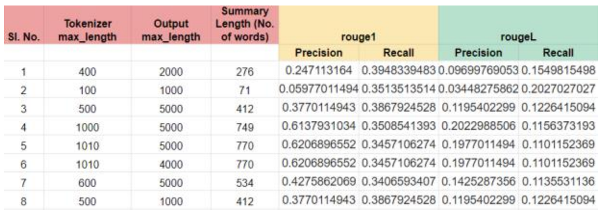


Sl. No.	Ratio Parameter	Summary Length (No. of Words)	rouge1		rougeL	
			Precision	Recall	Precision	Recall
1	0.2	397	0.6256	0.6287	0.4015	0.4035
2	0.3	566	0.625	0.662	0.3782	0.4007
3	0.5	916	0.7268	0.7761	0.5242	0.5597
4	0.6	1,100	0.7453	0.8079	0.5655	0.613
5	0.7	1,228	0.7568	0.8436	0.6151	0.6856

Fig. 3. BERT scores for Text-25

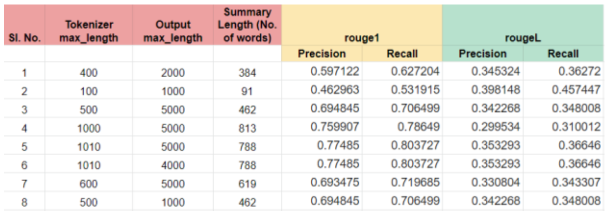
The percentage of sentences to be extracted from the data to generate the summary was varied. The lowest ROUGE-1 and ROUGE-L scores are obtained when 20% of the text is included in the summary, giving us a low 300-word summary. The largest summary size is obtained when parameters are tuned to 0.7, displaying the highest values of the ROUGE score.

There is a direct relation between length of summary and ROUGE score values. Since it is counterproductive to tune the model to obtain longer summaries, the optimal parameters for the BERT model will be ratio parameter equal to 0.3. The output length is highly dependent on the input length, resulting in a moderate ROUGE score.



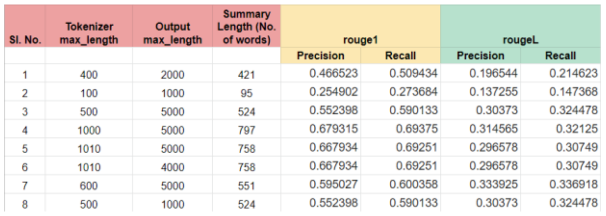
Sl. No.	Tokenizer max_length	Output max_length	Summary Length (No. of words)	rouge1		rougeL	
				Precision	Recall	Precision	Recall
1	400	2000	276	0.247113164	0.3948339483	0.09699769053	0.1549815498
2	100	1000	71	0.05977011494	0.3513513514	0.03448275862	0.2027027027
3	500	5000	412	0.3770114943	0.3867924528	0.1195402299	0.1226415094
4	1000	5000	749	0.6137931034	0.3508541393	0.2022988506	0.1156373193
5	1010	5000	770	0.6206896552	0.3457106274	0.1977011494	0.1101152369
6	1010	4000	770	0.6206896552	0.3457106274	0.1977011494	0.1101152369
7	600	5000	534	0.4275862069	0.3406593407	0.1425287356	0.1135531136
8	500	1000	412	0.3770114943	0.3867924528	0.1195402299	0.1226415094

Fig. 4. BART scores for Text-0



Sl. No.	Tokenizer max_length	Output max_length	Summary Length (No. of words)	rouge1		rougeL	
				Precision	Recall	Precision	Recall
1	400	2000	384	0.597122	0.627204	0.345324	0.36272
2	100	1000	91	0.462963	0.531915	0.398148	0.457447
3	500	5000	462	0.694845	0.706499	0.342268	0.348008
4	1000	5000	813	0.759907	0.78649	0.299534	0.310012
5	1010	5000	788	0.77485	0.803727	0.353293	0.36646
6	1010	4000	788	0.77485	0.803727	0.353293	0.36646
7	600	5000	619	0.693475	0.719685	0.330804	0.343307
8	500	1000	462	0.694845	0.706499	0.342268	0.348008

Fig. 5. BART scores for Text-7



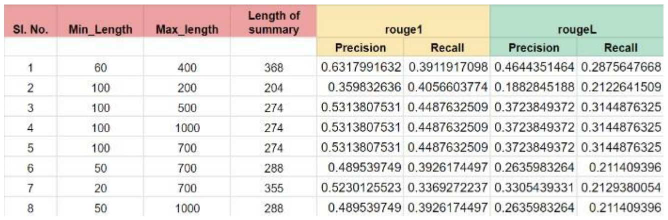
Sl. No.	Tokenizer max_length	Output max_length	Summary Length (No. of words)	rouge1		rougeL	
				Precision	Recall	Precision	Recall
1	400	2000	421	0.466523	0.509434	0.196544	0.214623
2	100	1000	95	0.254902	0.273684	0.137255	0.147368
3	500	5000	524	0.552398	0.590133	0.30373	0.324478
4	1000	5000	797	0.679315	0.69375	0.314565	0.32125
5	1010	5000	758	0.667934	0.69251	0.296578	0.30749
6	1010	4000	758	0.667934	0.69251	0.296578	0.30749
7	600	5000	551	0.595027	0.600358	0.333925	0.336918
8	500	1000	524	0.552398	0.590133	0.30373	0.324478

Fig. 6. BART scores for Text-25

The number of tokens tokenized from the data and the number of tokens the model takes as input have been varied.

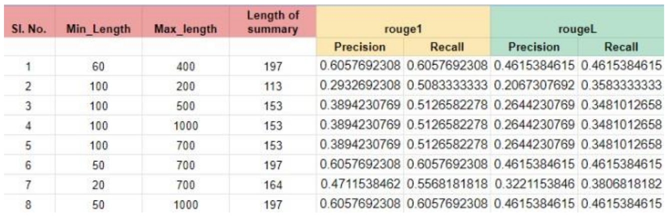
The lowest scores of ROUGE-1 and ROUGE-L are the result of 100 and 1000 being tokenizer max length and output max length respectively. The length of the summary is also minimal. The largest summary size is obtained when parameters are tuned to lengths 1010 and 5000, displaying the highest values of the ROUGE score.

There is a direct correlation between length of summary and ROUGE score values. Since it is counterproductive to tune the model to obtain longer summaries, the optimal parameters for the BART model will be tokenizer length equal to 600 and output length equal to 5000, resulting in a moderate ROUGE score.



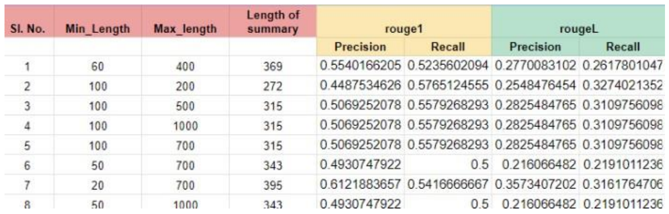
Sl. No.	Min_Length	Max_length	Length of summary	rouge1		rougeL	
				Precision	Recall	Precision	Recall
1	60	400	368	0.6317991632	0.3911917098	0.4644351464	0.2875647668
2	100	200	204	0.359832636	0.4056603774	0.1882845188	0.2122641509
3	100	500	274	0.5313807531	0.4487632509	0.3723849372	0.3144876325
4	100	1000	274	0.5313807531	0.4487632509	0.3723849372	0.3144876325
5	100	700	274	0.5313807531	0.4487632509	0.3723849372	0.3144876325
6	50	700	288	0.489539749	0.3926174497	0.2635983264	0.211409396
7	20	700	355	0.5230125523	0.3369272237	0.3305439331	0.2129380054
8	50	1000	288	0.489539749	0.3926174497	0.2635983264	0.211409396

Fig. 7. GPT-2 scores for Text-0



Sl. No.	Min_Length	Max_length	Length of summary	rouge1		rougeL	
				Precision	Recall	Precision	Recall
1	60	400	197	0.6057692308	0.6057692308	0.4615384615	0.4615384615
2	100	200	113	0.2932692308	0.5083333333	0.2067307692	0.3583333333
3	100	500	153	0.3894230769	0.5126582278	0.2644230769	0.3481012658
4	100	1000	153	0.3894230769	0.5126582278	0.2644230769	0.3481012658
5	100	700	153	0.3894230769	0.5126582278	0.2644230769	0.3481012658
6	50	700	197	0.6057692308	0.6057692308	0.4615384615	0.4615384615
7	20	700	164	0.4711538462	0.5568181818	0.3221153846	0.3806818182
8	50	1000	197	0.6057692308	0.6057692308	0.4615384615	0.4615384615

Fig. 8. GPT-2 scores for Text-7

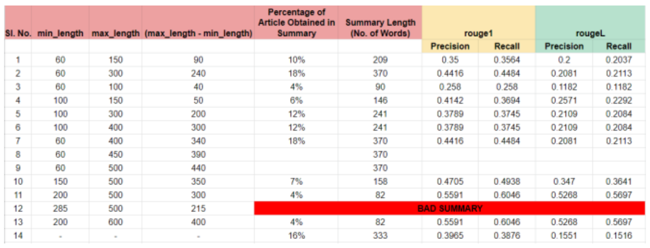


Sl. No.	Min_Length	Max_length	Length of summary	rouge1		rougeL	
				Precision	Recall	Precision	Recall
1	60	400	369	0.5540166205	0.5235602094	0.2770083102	0.2617801047
2	100	200	272	0.4487534626	0.5765124555	0.2548476454	0.3274021352
3	100	500	315	0.5069252078	0.5579268293	0.2825484765	0.3109756098
4	100	1000	315	0.5069252078	0.5579268293	0.2825484765	0.3109756098
5	100	700	315	0.5069252078	0.5579268293	0.2825484765	0.3109756098
6	50	700	343	0.4930747922	0.5	0.216066482	0.2191011236
7	20	700	395	0.6121883657	0.5416666667	0.3573407202	0.3161764706
8	50	1000	343	0.4930747922	0.5	0.216066482	0.2191011236

Fig. 9. GPT-2 scores for Text-25

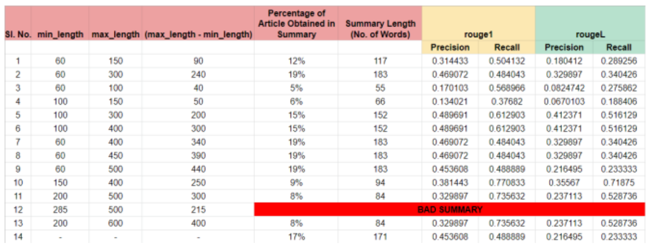
GPT-2 parameters are fine-tuned based on the minimum and maximum length of the output summaries. The poorest performing version has a minimum and maximum length of 100 and 200 respectively, with the resulting summary length of 1322. The ROUGE score improves with tuned parameters of 60 and 400, resulting in the largest summary length of 2030. The highest ROUGE scores are recorded with a minimum and maximum of 20 and 700 respectively, with the third highest summary length of 1903.

The value of the ROUGE scores are not proportional to the length of the summary obtained. Hence for the optimal parameters of the GPT-2 model, minimum length is fixed at 50, while maximum length may be set as 700 or 1000. They produce the second highest ROUGE scores and output the largest summary length as well, giving us 18% of the original text.



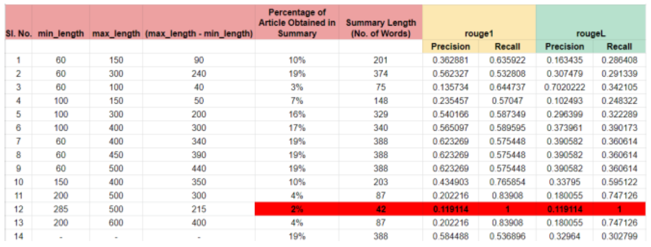
Sl. No.	min_length	max_length	(max_length - min_length)	Percentage of Article Obtained in Summary	Summary Length (No. of Words)	rouge1		rougeL	
						Precision	Recall	Precision	Recall
1	60	150	90	10%	209	0.35	0.3564	0.2	0.2037
2	60	300	240	18%	370	0.4416	0.4484	0.2081	0.2113
3	60	100	40	4%	90	0.258	0.258	0.1182	0.1182
4	100	150	50	6%	146	0.4142	0.3694	0.2571	0.2292
5	100	300	200	12%	241	0.3789	0.3745	0.2109	0.2084
6	100	400	300	12%	241	0.3789	0.3745	0.2109	0.2084
7	60	400	340	18%	370	0.4416	0.4484	0.2081	0.2113
8	60	450	390		370				
9	60	500	440		370				
10	150	500	350	7%	158	0.4705	0.4938	0.347	0.3641
11	200	500	300	4%	82	0.5591	0.6046	0.5268	0.5697
12	285	500	215	BAD SUMMARY					
13	200	600	400	4%	82	0.5591	0.6046	0.5268	0.5697
14	-	-	-	16%	333	0.3965	0.3876	0.1551	0.1516

Fig. 10. XLNet scores for Text-0



Sl. No.	min_length	max_length	(max_length - min_length)	Percentage of Article Obtained in Summary	Summary Length (No. of Words)	rouge1		rougeL	
						Precision	Recall	Precision	Recall
1	60	150	90	12%	117	0.314433	0.504132	0.180412	0.289256
2	60	300	240	19%	183	0.469072	0.484043	0.329897	0.340426
3	60	100	40	5%	55	0.170103	0.568966	0.0824742	0.275862
4	100	150	50	6%	66	0.134021	0.37682	0.0670103	0.188406
5	100	300	200	15%	152	0.489691	0.612903	0.412371	0.516129
6	100	400	300	15%	152	0.489691	0.612903	0.412371	0.516129
7	60	400	340	19%	183	0.469072	0.484043	0.329897	0.340426
8	60	450	390	19%	183	0.469072	0.484043	0.329897	0.340426
9	60	500	440	19%	183	0.453608	0.488889	0.216495	0.233333
10	150	400	250	9%	94	0.381443	0.770833	0.35567	0.71875
11	200	500	300	8%	84	0.329897	0.735632	0.237113	0.528736
12	285	500	215	BAD SUMMARY					
13	200	600	400	8%	84	0.329897	0.735632	0.237113	0.528736
14	-	-	-	17%	171	0.453608	0.488889	0.216495	0.233333

Fig. 11. XLNet scores for Text-7



Sl. No.	min_length	max_length	(max_length - min_length)	Percentage of Article Obtained in Summary	Summary Length (No. of Words)	rouge1		rougeL	
						Precision	Recall	Precision	Recall
1	60	150	90	10%	201	0.362881	0.635922	0.163435	0.286408
2	60	300	240	19%	374	0.562327	0.532808	0.307479	0.291339
3	60	100	40	3%	75	0.135734	0.644737	0.7020222	0.342105
4	100	150	50	7%	148	0.235457	0.57047	0.102493	0.248322
5	100	300	200	16%	329	0.540166	0.587349	0.296399	0.322289
6	100	400	300	17%	340	0.565097	0.589595	0.373961	0.390173
7	60	400	340	19%	388	0.623269	0.575448	0.390582	0.360614
8	60	450	390	19%	388	0.623269	0.575448	0.390582	0.360614
9	60	500	440	19%	388	0.623269	0.575448	0.390582	0.360614
10	150	400	350	10%	203	0.434903	0.765854	0.33795	0.595122
11	200	500	300	4%	87	0.202216	0.83908	0.180055	0.747126
12	285	500	215	2%	42	0.119114	1	0.119114	1
13	200	600	400	4%	87	0.202216	0.83908	0.180055	0.747126
14	-	-	-	19%	388	0.584488	0.536896	0.32964	0.302799

Fig. 12. XLNet scores for Text-25

XLNet parameters are fine-tuned based on the minimum and maximum length of the output summaries. The poorest performing version has a minimum and maximum length of 60 and 100 respectively, with the resulting summary length of 90 and a 4% text inclusion in the output. The score improves with tuned parameters of 200 and 600, resulting in the largest summary length of 400 and obtaining 4% of the article in the summary.

The value of the ROUGE scores are evidently not proportional to the length of the summary obtained. Hence, for the optimal parameters of the XLNet model, minimum length is fixed at 60, while maximum length may be set as 300 or 400. They produce the third-highest scores, output the largest summary length, and include the highest percentage of the text in the output, which is 18%.

HYBRID MODELS

In an effort to optimise the results obtained from these four models, a hybrid approach was attempted. The following combinations were implemented, and the results prove that a hybrid approach leads to better scores. In addition, we have also tried the combinations in two different ways, alternating between the order in which the models are implemented in the article.

TABLE III
HYBRID MODEL TESTING ON TEXT - 0

Order of Models	Precision		Recall	
	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L
BERT + BART	0.9075	0.7797	0.2276	0.1955
BART + BERT	0.7621	0.3877	0.1367	0.0695
BERT + GPT-2	0.8634	0.6960	0.2341	0.1888
GPT-2 + BERT	0.7180	0.4625	0.2345	0.1511
XLNET + GPT-2	0.4493	0.2907	0.4286	0.2773
GPT-2 + XLNET	0.7621	0.3877	0.1367	0.0695
XLNET + BERT	0.6828	0.3833	0.2618	0.1469
BERT + XLNET	0.8502	0.6872	0.2497	0.2018

TABLE IV
HYBRID MODEL TESTING FOR TEXT - 7

Order of Models	Precision		Recall	
	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L
BERT + BART	0.7950	0.5237	0.8060	0.5310
BART + BERT	0.8105	0.5338	0.8276	0.5451
BERT + GPT-2	0.6450	0.4916	0.6987	0.5324
GPT-2 + BERT	0.6163	0.4652	0.7021	0.5300
XLNET + GPT-2	0.6755	0.5763	0.7254	0.6188
GPT-2 + XLNET	0.4549	0.2832	0.4796	0.2986
XLNET + BERT	0.6618	0.5107	0.7360	0.5680
BERT + XLNET	0.6019	0.4292	0.6765	0.4824

It is observed that the highest ROUGE scores are obtained from the combination of BERT + BART, followed by BERT + GPT-2. From this, we can deduce the following points -

- Individually, BERT has its shortcomings, which have been explored in further sections. However, when combined with other models, it boosts performance.
- BART works better individually than when combined with other models.
- XLNet and GPT-2 are not good performers when combined together.

B. *Individual Article Comparison*

Two texts from the dataset have been chosen for the comparison, namely Text-7 and Text-25.

1) **Text-7** is 995-words long, and is the shorter of the two. It almost completely consists of alphabetical characters, with very few numeric and scientific ones. It has many potential tokenizable words and is void of any special characters or equations. Model performance on this text is measured using ROUGE-1, due to its shortness of length.
The following graphs show how BERT fares against the other models according to ROUGE-1 scores.

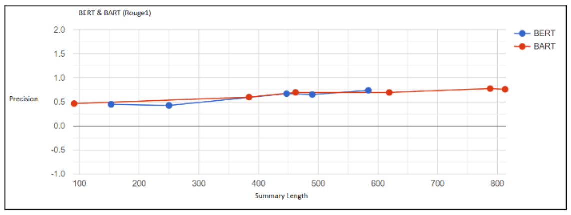


Fig. 13. BERT v/s BART performance according to ROUGE-1

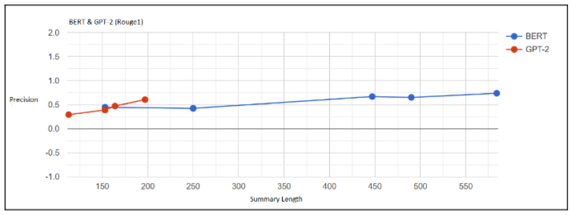


Fig. 14. BERT v/s GPT-2 performance according to ROUGE-1

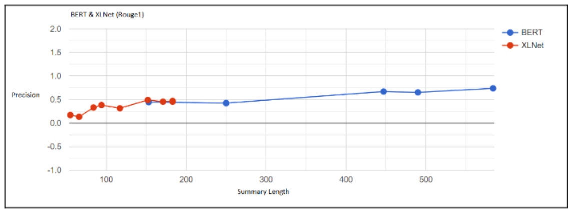


Fig. 15. BERT v/s XLNet performance according to ROUGE-1

These figures could indicate that length of summary is not an important parameter while considering the effectiveness of a summary of a comparatively short article. This is evidenced by the fact that although BERT's output might be longer than the other models', it is still regarded as the best in terms of ROUGE metrics.

2) **Text-25** is roughly 1941-words long. It is highly technical, containing various numeric characters and scientific formulae. Model performance on this text is measured using ROUGE-L, due to its extensive length and complexity.
The following graphs show how BART fares against other models according to ROUGE-L scores.

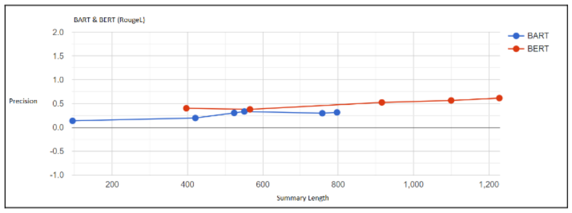


Fig. 16. BART v/s BERT performance according to ROUGE-L

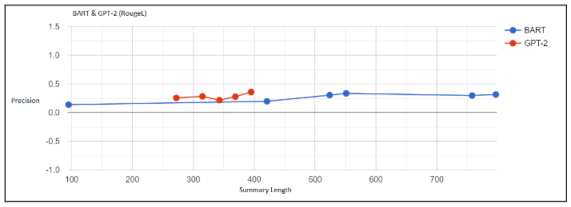


Fig. 17. BART v/s GPT-2 performance according to ROUGE-L

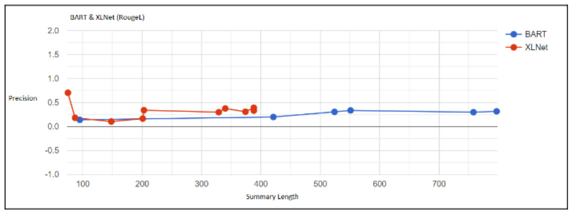


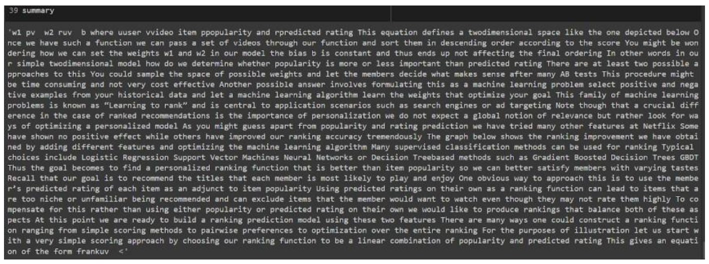
Fig. 18. BART v/s XLNet performance according to ROUGE-L

These figures indicate that length of the output summary plays a key role in the ROUGE metric output. Shorter summaries have obtained better scores than longer ones that include more information from the original text.

C. *Word2Vec*

The summary obtained by the Word2Vec algorithm ranks low on readability, despite demonstrating a reasonably high ROUGE score. It does not capture the important points of the original text, does not provide a cohesive output, and has multiple spelling errors.

The following is the output received on summarizing Text-25 using the Word2Vec algorithm. It is evident that as the size of the original text increases, the quality of the summary decreases.



```
39 summary

'w1 pv  w2 ruv  b where uuser vvideo item ppopularity and rpredicted rating This equation defines a twodimensional space like the one depicted below O
nce we have such a function we can pass a set of videos through our function and sort them in descending order according to the score You might be won
dering how we can set the weights w1 and w2 in our model the bias b is constant and thus ends up not affecting the final ordering In other words in ou
r simple twodimensional model how do we determine whether popularity is more or less important than predicted rating There are at least two possible a
pproaches to this You could sample the space of possible weights and let the members decide what makes sense after many AB tests This procedure might
be time consuming and not very cost effective Another possible answer involves formulating this as a machine learning problem select positive and nega
tive examples from your historical data and let a machine learning algorithm learn the weights that optimize your goal This family of machine learning
problems is known as “Learning to rank” and is central to application scenarios such as search engines or ad targeting Note though that a crucial diff
erence in the case of ranked recommendations is the importance of personalization we do not expect a global notion of relevance but rather look for wa
ys of optimizing a personalized model As you might guess apart from popularity and rating prediction we have tried many other features at Netflix Some
have shown no positive effect while others have improved our ranking accuracy tremendously The graph below shows the ranking improvement we have obtai
ned by adding different features and optimizing the machine learning algorithm Many supervised classification methods can be used for ranking Typical
choices include Logistic Regression Support Vector Machines Neural Networks or Decision Treebased methods such as Gradient Boosted Decision Trees GBDT
Thus the goal becomes to find a personalized ranking function that is better than item popularity so we can better satisfy members with varying tastes
Recall that our goal is to recommend the titles that each member is most likely to play and enjoy One obvious way to approach this is to use the membe
r’s predicted rating of each item as an adjunct to item popularity Using predicted ratings on their own as a ranking function can lead to items that a
re too niche or unfamiliar being recommended and can exclude items that the member would want to watch even though they may not rate them highly To co
mpensate for this rather than using either popularity or predicted rating on their own we would like to produce rankings that balance both of these as
pects At this point we are ready to build a ranking prediction model using these two features There are many ways one could construct a ranking functi
on ranging from simple scoring methods to pairwise preferences to optimization over the entire ranking For the purposes of illustration let us start w
ith a very simple scoring approach by choosing our ranking function to be a linear combination of popularity and predicted rating This gives an equati
on of the form frankuv  <'
```

Fig. 19. Word2Vec Summary of Text-25

The conclusions that have been arrived at from the above implementations of the models and algorithms are as follows -

1) The optimal parameters for the four pre-trained models are as follows -
   a) BERT - ratio = 0.3
   b) BART - tokenizer_max_length = 600 and output_-max_length = 5000
   c) GPT-2 - min_length = 60 and max_length = 400 or 1000
   d) XLNet - min_length = 60 and max_length = 300 or 400

2) Although the parameters for these models have been fine-tuned to obtain these results, it is important to note that these are not necessarily the highest ROUGE scores obtained. Taking into consideration that longer summaries generate *higher* scores, and summary length needs to be retained at 30% of the length of the original text, the parameters listed above are the *optimal* values that give reasonably high ROUGE scores.

3) The BERT model's performance decreases as the length of the original text increases. This could point to a potential drawback of the model itself, making it efficient for shorter articles but not as much for longer ones.

4) Despite the fact that XLNet's most important factor is its use of Transfomer-XL, removing long-term dependency, it can be seen that it performs poorly on both short and long texts. This is evident in its ROUGE scores. Although the model has been known to give SOTA results on other NLP tasks such as sentiment analysis, text classification, and document ranking, its extractive summarization capabilities are unclear.

5) The BART model has proven effective for both short and longform texts. This could potentially be due to the initial training of the model itself, where it had achieved SOTA results for the task of summarization particularly.

6) GPT-2's results have been average across text lengths. This could be a direct result of the initial task-agnostic training method employed.

7) The results from these four models could indicate that an encoder-only architecture (like BERT) or decoder-only model (like GPT-2) do not perform as well as an encoder-decoder model, like BART, as text size increases.

8) Hybrid models have been proven to be more effective when compared to individual performances. This could lead to further study by combining various language models in different combinations and ratios to obtain the best possible result.

9) Traditional algorithms like Word2Vec are not as efficient for NLP tasks, especially in the light of recent machine learning innovations. Although pre-trained language models are compared with each other in an effort to arrive at the one that obtains the best summary, it is clear that compared to legacy algorithms, machine learning based approaches perform far better in terms of readability, logic, and cohesiveness.

10) With regards to the evaluation of automatically generated summaries, metrics like ROUGE are insufficient to truly achieve a user-targeted score. For practical purposes, it is necessary to either carry out user studies to accurately judge the effectiveness of a generated summary, or to develop a metric that is capable of critiquing the obtained summary from a human perspective. It is inadequate to assess the quality of a summary using only parameters such as word count, spellings, commonality with original text, or grammar. For a truly well-rounded output, it is necessary to also look at the summary from abstract evaluators such as readability, chronology, and whether it captures the important aspects of the original. Subjectivity is a necessary aspect of judging summary outputs.

IX. Conclusion and Further Study

This research has spanned various topics in the field of extractive automatic summarization of articles. Modern methods like pre-trained language models have been analysed and implemented. In an effort to obtain optimal results as per the ROUGE metric, modifications to parameters of individual models, as well as combinations of the best performing models have been carried out. In addition, legacy techniques like Word2Vec have also been examined, and verified using the same method.

The conclusions of the investigation, as enumerated in the previous section, indicate that the four models have varying degrees of success with extractive summarization. Their individual performances indicate that they are better suited for different specifications of the task. They also show that the initial dataset and manner of pre-training play a large role in their capability.

The implementation of the hybrid models prove that when combined, the outputs obtained are drastically better than their individual working. This indicates a potential method for improved results across NLP tasks, and a field open for further research.

The Word2Vec algorithm pales in the light of modern innovations, its results revealing its comparatively mediocre performance, unsuited for quintessential NLP activities.

Utilisation of ROUGE metrics, seemingly a staple of recent summarization research, has proven to be inadequate in true judgement of a model summary. Results obtained from empirical methods do not necessarily align with those that are human-evaluated. For a truly sophisticated and informed summary, a multi-faceted judgement is necessary.

This research has covered a range of topics in the field of extractive summarization, including the performance of pre-trained language models, traditional algorithms, and machine evaluation of results. It begets several new questions and proposes novel potential fields of study, including the efficiency of quantitative evaluation methods and the benefits of a hybrid modelling technique.

Pre-trained language models are an efficient way to harness machine learning novelty to address essentially NLP tasks. However, within the same field, there are various modifications that can be made to better the outputs obtained. These modifications can be done on individual models, to get the best possible summary using just a single language model. One can also attempt to combine various models and test their amalgamated performance. Apart from these tweaks to the existing infrastructure, there is room for new models that could use different parameters.

Recent innovations for the task of automatic text summarization include graph-based methods, deep learning approaches, and combining extractive and abstractive techniques for optimal output summaries. Although in their fledgling state of research, all of these practices promise greater efficiency and more precise results in the eventual future.

REFERENCES

[1] Cambridge Dictionary, "summarization," @CambridgeWords, Nov. 30, 2022. https://dictionary.cambridge.org/dictionary/english/summarization (accessed Dec. 04, 2022).

[2] "Text Compactor: Free Online Automatic Text Summarization Tool," Textcompactor.com, 2010. https://www.textcompactor.com/

[3] J. Devlin, M.-W. Chang, K. Lee, K. Google, and A. Language, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," 2019. [Online]. Available:https://arxiv.org/pdf/1810.04805.pdf

[4] H. P. Luhn, "The Automatic Creation of Literature Abstracts," IBM Journal of Research and Development, vol. 2, no. 2, pp. 159–165, 1958, doi: 10.1147/rd.22.0159.

[5] S. Alhojely and J. Kalita, "Recent Progress on Text Summarization," 2020 International Conference on Computational Science and Computational Intelligence (CSCI), 2020, [Online]. Available: https://american-cse.org/sites/csci2020proc/pdfs/CSCI2020-6SccvdzjqC7bKupZxFmCoA/762400b503/762400b503.pdf

[6] J.-P. Ng and V. Abrecht, "Better Summarization Evaluation with Word Embeddings for ROUGE," arXiv:1508.06034 [cs], Aug. 2015, Accessed: Dec. 04, 2022. [Online]. Available: https://arxiv.org/abs/1508.06034v1

[7] M. Peyrard, T. Botschen, and I. Gurevych, "Learning to Score System Summaries for Better Content Selection Evaluation.," ACLWeb, Sep. 01, 2017. https://aclanthology.org/W17-4510/(accessedDec.04,2022).

[8] Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, Dragomir Radev; SummEval: Re-evaluating Summarization Evaluation. Transactions of the Association for Computational Linguistics 2021; 9 391–409. doi: https://doi.org/10.1162/tacl_a_00373

[9] O. Vasilyev, V. Dharnidharka, and J. Bohannon, "Fill in the BLANC: Human-free quality estimation of document summaries," arXiv:2002.09836 [cs], Nov. 2020, Accessed: Dec. 04, 2022. [Online]. Available: https://arxiv.org/abs/2002.09836#:~:text=Fill%20in%20the%20BLANC%3A%20Human%2Dfree%20quality%20estimation%20of%20document%20summaries\T1\textcompwordmark

[10] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating Text Generation with BERT," arXiv:1904.09675 [cs], Feb. 2020, [Online]. Available: https://arxiv.org/abs/1904.09675

[11] Medium, "Medium," Medium, 2018. https://medium.com/

[12] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," aclanthology.org, Jul. 01, 2004. https://aclanthology.org/W04-1013/

[13] G. LLC, "rouge-score: Pure python implementation of ROUGE-1.5.5.," PyPI. https://pypi.org/project/rouge-score/#:~:text=ROUGE%20for%20Python (accessed Dec. 04, 2022).

[14] Liu, Yang, and Mirella Lapata. "Text Summarization with Pretrained Encoders." ArXiv:1908.08345 [Cs], Sept. 2019. arXiv.org, http://arxiv.org/abs/1908.08345.

[15] Alomari, Ayham, et al. "Deep Reinforcement and Transfer Learning for Abstractive Text Summarization: A Review." Computer Speech & Language, vol. 71, Jan. 2022, p. 101276. DOI.org (Crossref), https://doi.org/10.1016/j.csl.2021.101276.

[16] Vaswani, Ashish, et al. "Attention Is All You Need." ArXiv:1706.03762 [Cs], Dec. 2017. arXiv.org, http://arxiv.org/abs/1706.03762.

[17] Dai, Zihang, et al. "Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context." ArXiv:1901.02860 [Cs, Stat], June 2019. arXiv.org, http://arxiv.org/abs/1901.02860.

[18] Yang, Zhilin, et al. "XLNet: Generalised Autoregressive Pre Training for Language Understanding." ArXiv:1906.08237 [Cs], Jan. 2020. arXiv.org, http://arxiv.org/abs/1906.08237.

[19] Lewis, Mike, et al. "BART: Denoising Sequence-to-Sequence Pre-Training for Natural Language Generation, Translation, and Comprehension." Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2020, pp. 7871–80. DOI.org (Crossref)

[20] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, 'Language Models are Unsupervised Multitask Learners', 2019.

[21] "Understanding XLNet - Borealis AI." https://www.borealisai.com/research-blogs/understanding-xlnet/

[22] "Pytorch-Transformers — pytorch-transformers 1.0.0 documentation," huggingface.co. https://huggingface.co/transformers/v1.2.0/

[23] T. P. Alvin, “Introduction to Text Summarization with ROUGE Scores,” Medium, Mar. 25, 2022. https://towardsdatascience.com/introduction-to-text-summarization-with-rouge-scores-84140c64b471

[24] “Medium Articles,” www.kaggle.com. https://www.kaggle.com/datasets/hsankesara/medium-articles

Biography

Jigisha M Narrain is a final-year B.Tech student at PES University, Bangalore. She is pursuing her bachelor’s degree in Computer Science and Engineering. She is interested in artificial intelligence, database management, and data science.

Vanshika Taneja is a final-year B.Tech student at PES University, Bengaluru. She is pursuing her bachelor’s degree in Computer Science and Engineering. She is interested in data science and machine learning algorithms.

Sanjana B Atrey is a final-year B.Tech student at PES University, Bengaluru. She is pursuing her bachelor’s degree in Computer Science and Engineering. She is interested in artificial intelligence, programming languages and web development.

Jahnavi Sivaram is a final-year B.Tech student at PES University, Bangalore. She is pursuing her bachelor’s degree in Computer Science and Engineering. She is interested in artificial intelligence and data analytics.

Dinesh Singh is an associate professor in the Department of Computer Science at PES University. His areas of interest include data structures, algorithms, machine learning.