FACTA UNIVERSITATIS
Series: **Electronics and Energetics** Vol. 37, Nº 2, June 2024, pp. 261-276
https://doi.org/10.2298/FUEE2402261S

Original scientific paper

DEEP LEARNING-BASED MODIFIED TRANSFORMER MODEL FOR AUTOMATED NEWS ARTICLE SUMMARIZATION

B. Srinivas[1], Lavanya Bagadi[1], Naresh K. Darimireddy[2], P. Surya Prasad[1], Sivaji Satrupalli[3], Anil Kumar B.[4]

[1]MVGR College of Engineering (A), Vizianagaram, India-535005
[2*]Lendi Institute of Engineering & Technology (A), Vizianagaram, India-535005
[3]Vignan's F S T R (Deemed to be University), Guntur, India-522213.
[4]GMR Institute of Technology (A), Rajam, India -532127

ORCID iDs:	B. Srinivas	https://orcid.org/0000-0002-9562-2325
	Lavanya Bagadi	https://orcid.org/0000-0002-0310-2411
	Naresh K. Darimireddy	https://orcid.org/0000-0002-0033-716X
	P. Surya Prasad	https://orcid.org/0000-0002-2767-123X
	Sivaji Satrupalli	https://orcid.org/0000-0002-9319-0493
	Anil Kumar B.	https://orcid.org/0000-0002-3468-3650

Abstract. *The amount of textual data on the internet is increasing enormously, so data summarization into text has become essential. As generating text summaries manually is an arduous task and humans are generally prone to make mistakes, deep learning techniques have evolved to overcome this problem. Modified transformer-based deep learning models with varying encoder-decoder and feed-forward network layers are proposed to develop an abstractive summary of the news articles. The proposed transformer model provides the advantage of parallelization with the help of multiple attention head layers to process long sentences, and hence, better text summarization performance is achieved. These models are trained on an 'in-shorts' dataset, and the proposed model is compared with the PEGASUS-CNNdaily-mail, BART-large-CNN, and DistilBART-CNN-12-6 models on the CNN/DailyMail dataset. The performance is evaluated in terms of the ROUGE score by comparing it with the existing Recurrent Neural Network (RNN) model. The suggested transformer model achieved a ROUGE score of 0.33, surpassing the RNN model score of 0.17. This innovative approach can be employed on extensive textual data to extract summaries or headlines.*

Key words: *Natural Language Processing, Deep Learning, Abstractive summarization, Large Language Model, RNN, Transformer Model, News Article Summarization*

Received June 02, 2023; revised December 15, 2023 and December 30, 2023; accepted January 09, 2024

Corresponding author: Naresh K. Darimireddy
Lendi Institute of Engineering & Technology (A), Vizianagaram, India-535005.
E-mail: yosuna@ieee.org

1. INTRODUCTION

In recent years, internet usage has increased, and without using any physical means, everything is being uploaded on the internet for a faster spread. Due to the increase in data that is being uploaded heavily into the internet, storage has become a problem. The earlier usage of uploading data for summarization was done manually, and it involves human mistakes and is a time-consuming process. Using deep learning to summarize and upload the NEWS data to the internet saves time. It reduces the space occupied for the data to generate a summary. Due to the explosive growth of textual data [1], attention to developing informative summaries using automatic methods is extensively trending. In today's busy schedule, people have limited time to read lengthy texts, making investing time in superfluous information impractical. To overcome this problem, automation can filter misinformation from critical information, but it is challenging. Extractive summaries used reinforcement learning-based, recurrent neural network-based encoders, and seq2seq encoder-decoder models to address this. However, with the advancement of deep learning models like Transformer and RNN, the earlier approaches used to generate abstractive summaries, including structure-based ones like graphs, trees, and ontology-based, etc., have become obsolete. Summarization condenses extensive information into a concise form that captures the central theme or idea. Text summarization primarily aims to transform lengthy texts into shorter yet meaningful representations. Presently, abstractive summaries are advancing in producing fluent and flexible summaries.

Further, Text summarization can serve many purposes, including generating email summaries, condensing news headlines, summarizing movie reviews, outlining student notes, and providing concise information for government officials and business professionals. It is also instrumental in translating medical data for doctors, condensing legal documents, generating novel or book summaries, and assisting consumers in deciding whether to read the content. It also plays a role in code summarization. Automatic summarization is classified into abstractive and extractive [2]. This work is mainly focused on the abstractive approach.

Extractive summarization generates the summary by considering the score from the article, i.e., the words or phrases are directly taken from the article and are joined to get an overview. Alternatively, abstractive summarization analyses the sentences and reconstructs the summary with a meaningful sentence structure. Thus, the abstractive outline is more flexible. The main contributions of this work are:

1) Text summarization through RNN is implemented as an introductory model, and its performance is calculated to get a proper understanding of summarization in deep learning.

2) Improvement in the performance of text summarization with the proposed model using modified transformers with varying numbers of encoder-decoder and feed-forward network layers.

3) The proposed model is compared with existing abstractive text summarization models such as PEGASUS-CNN_dailymail, BART-large-CNN, and DistilBART-CNN-12-6 on the CNN/Dailymail dataset.

1.1. Related Work

A survey on text summarization reveals the key findings from the literature in this field to address the challenge of condensing large volumes of text into concise and coherent summaries. The following summarizes the researcher's exploration of various approaches and techniques.

Research has focused on two main approaches, namely extractive and abstractive. Extractive involves selecting and arranging existing sentences, whereas abstractive generates new sentences to convey the essence of the text. Integrating machine learning and NLP techniques is employed in advancing text summarization to create summaries for supervised and unsupervised learning models. The rise of deep learning [3] has led to the application of neural network models such as Recurrent Neural Networks (RNNs) [4], Long Short-Term Memory networks (LSTMs), [5] and Transformer models like BERT and GPT [6] in text summarization tasks. Various researchers evaluate the quality of summaries by including metrics like ROUGE (Recall-Oriented Understudy for Gisting Evaluation) and BLEU (Bilingual Evaluation Understudy). These metrics aim to measure the overlap and fluency of generated summaries compared to reference summaries. The literature addresses challenges in exploring various techniques, including cluster-based summarization and graph-based models, to handle the complex task of summarizing information from multiple documents. Some studies focus on domain-specific outlines, tailoring approaches to the unique characteristics and requirements of particular fields such as biomedical literature, legal documents, or news articles. Recent efforts have been directed towards developing real-time summarization systems to cater to the needs of summarizing information based on the dynamic nature of news and social media.

Among the summaries employed (abstractive and extractive), the need for abstractive summaries arises due to the limitations of extractive summaries. Extractive summaries must generate coherent text and merely concatenate extracted content without verifying grammar to reproduce meaningful text. Numerous abstractive summarization models have been introduced in the existing literature. The literature survey [7] highlights various approaches such as graph-based [8], rule-based [9], and semantic modeling [10]. Nevertheless, these earlier models fail to leverage recent breakthroughs in deep learning, which have demonstrated enhancements in numerous Natural language Processing (NLP) tasks.

Text summarization using NLP appeared in 1958. At first, statistical approaches were used to extract the most significant words in the sentences. Then, according to the score assigned to the ruling, the highest score is considered, and a summary is generated. The process mentioned above is for extractive summarization. Though they can produce a resume, the process is just trimming the original text, which is not an effective way of summarizing. The model has to make the summary, which must be close to the outline generated by an expert.

In contrast, the abstractive summarization methods [11] are proposed, which reads and understands the meaning of the source text or news article and then reproduces the summary meaningfully.

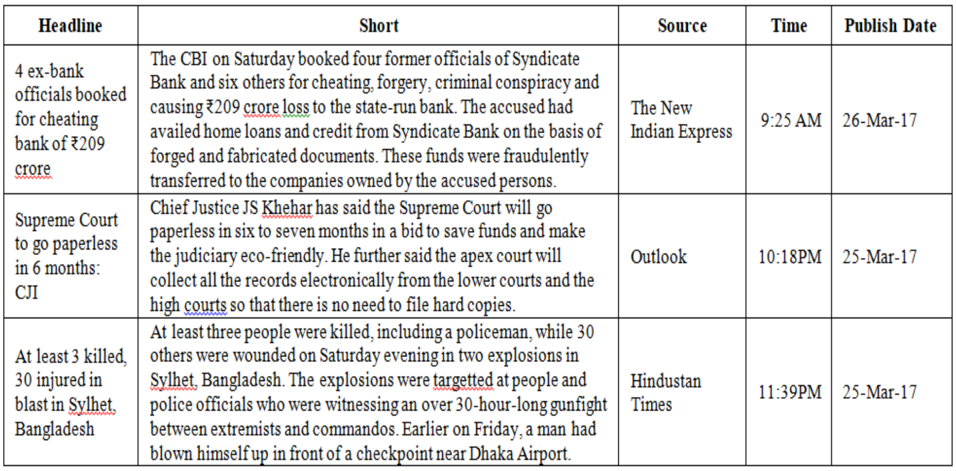
Chen et al. [12] suggested a multi-step procedure based on compression-paraphrase to rewrite a document by extracting relevant sentences for abstractive text summarization. An actor-critic algorithm was used in their model to optimize the sentence extractor and improve extraction. Li et al. [13] presented an actor-critic method to distinguish the ground truth from generated summaries using a binary classifier neural network. The actor network consists of an attention-based seq2seq model and a critic network with a global summary quality estimator and a maximum likelihood estimator. Zhang et al. [14] recommended a curriculum learning strategy and a Reinforced algorithm to simplify the sentences. Lin et al. [15] also summarized long documents using the coarse-to-fine attention method. Pasunuru et al. [16] trained the pointer-generator network and introduced two new metrics, saliency and entailment, and the ROUGE score through the self-critic policy gradient algorithm.

Kalchbrenner et al. [17] presented a Byte Net model to the encoder and decoder with a fixed depth for both by introducing a convolution neural network [18].

The Transformer [19] is a deep learning architecture composed of a sequence of encoder and decoder layers. These layers employ the attention mechanism to capture global dependencies within sequential data [20]. Vaswani et al. [19] developed a new architecture called the Transformer model, which depends on a multi-head attention mechanism and feed-forward network. Lewis et al. [21] used transformer architecture with pre-trained models on a massive amount of text for summarization purposes.

2. DATASET

The dataset used for this work is ‘in-shorts’ news data. It can be used for various tasks, such as text summarization, sentiment analysis, topic modeling, etc. It consists of five columns with unique values such as headline, short (news), source, time, and publish date [22], as shown in Fig. 1. The headline refers to the headline given by an expert, and the short refers to the actual news article. To perform text summarization, the headline column as the target summary and the short column as the source text are considered, and then the performance of different models is compared with the expert-written headlines in the dataset. The total number of samples in the dataset is 55,104, with a training set consisting of 53 K samples to fit the model and a test set of 2,104 samples to evaluate the model. The pre-processing steps are initiated after dropping the dataset's source, time, and publish date column values.



Headline	Short	Source	Time	Publish Date
4 ex-bank officials booked for cheating bank of ₹209 crore	The CBI on Saturday booked four former officials of Syndicate Bank and six others for cheating, forgery, criminal conspiracy and causing ₹209 crore loss to the state-run bank. The accused had availed home loans and credit from Syndicate Bank on the basis of forged and fabricated documents. These funds were fraudulently transferred to the companies owned by the accused persons.	The New Indian Express	9:25 AM	26-Mar-17
Supreme Court to go paperless in 6 months: CJI	Chief Justice JS Khehar has said the Supreme Court will go paperless in six to seven months in a bid to save funds and make the judiciary eco-friendly. He further said the apex court will collect all the records electronically from the lower courts and the high courts so that there is no need to file hard copies.	Outlook	10:18PM	25-Mar-17
At least 3 killed, 30 injured in blast in Sylhet, Bangladesh	At least three people were killed, including a policeman, while 30 others were wounded on Saturday evening in two explosions in Sylhet, Bangladesh. The explosions were targetted at people and police officials who were witnessing an over 30-hour-long gunfight between extremists and commandos. Earlier on Friday, a man had blown himself up in front of a checkpoint near Dhaka Airport.	Hindustan Times	11:39PM	25-Mar-17

Fig. 1 Sample snippet of in-short database

3. METHODOLOGY

The block diagram for the abstractive summarization process using a transformer-based deep learning model is shown in Fig. 2. The input article is taken; pre-processing is performed on the data, followed by positional encoding and masking [23, 24]. Now, the tokenized and encoded text is given to the layers present in the model to train the model. Then, the relationship between the input articles and the targets (Headlines or Summary) can be defined. Then, testing data is passed onto the model to obtain its summary or headlines.

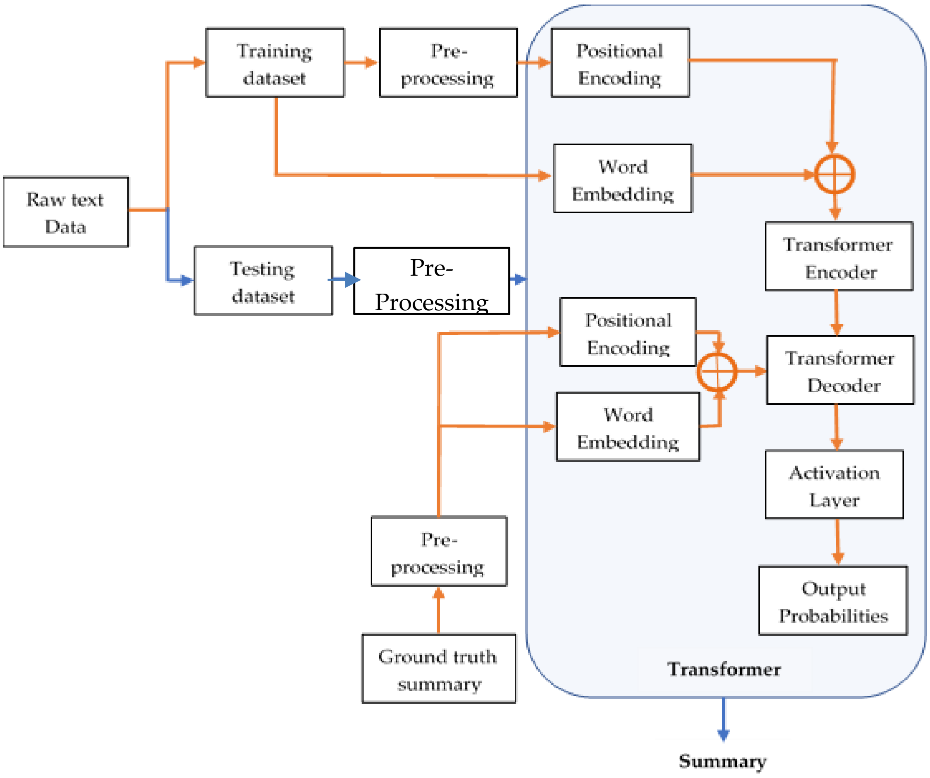


Fig. 2 Abstractive Summarization Methodology

3.1. Pre-processing

The dataset is given as an input to the model after the pre-processing step in any NLP task. Data pre-processing is done in three stages: tokenization, padding and truncating, and batching and shuffling. In tokenization, the text sequence removes punctuation marks and is converted to lowercase. Internally, a vocab dictionary is present, arranged in the order of frequency by which the words occur, and then it maps the words into respective tokens. Tokens are numerical representations of the sequence of terms. The padding and truncating step helps maintain the input's fixed length at the encoder. Padding refers to adding unique tokens to the information with fewer tokens, whereas truncating helps reduce the tokens if more tokens are present at the input. Batching and shuffling are performed for easy fetching of data. Here, the data sent to the model is batched instead of passing as a single value.

Two utility functions are performed on the cleaned dataset, and these functions are present in the model. They are positional encoding and masking. In NLPtasks like summarization, the order of words is essential as the meaning given by the sentence entirely changes with the order of words. So, the order of words is preserved through positional encoding. Masking is mainly used in transformers to prevent the terms present after the current word from involving those in the prediction of the current word.

The model is developed for encoding and decoding; then, the dataset is cleaned and sent as input to the model after positional encoding. The dataset is prepared to train the model, and pre-processing is performed on the testing dataset. After the prediction of the summary of test inputs, the ROUGE metric [25] is calculated for the testing data.

3.2. Transformer Model

The sequential problem faced by most of the NLP algorithms can be overcome using a transformer model, as it uses parallelization with the help of multiple attention head layers while processing the input text. Thus, the transformer model provides the advantage of processing long sentences in less time [26, 27].

The architecture of the transformer model is an encoder-decoder structure, and for normalizing the output probabilities, the Softmax activation function is used at the end. The model takes a sequence of data as input. Then, the embedded input words are assigned vectors based on the positioning of words in a sentence before passing through the positional encoders [21, 28].

The embeddings are received by the encoder blocks consisting of multi-head attention and feed-forward network paths, as shown in Fig. 2. The relationship between the words is captured in a sentence using multi-head attention layers to compute the attention vectors for each input and to represent how each word is related to other words in the same sentence. The decoder block uses attention vectors with one vector at a time in a feed-forward network. In the multi-head attention layer, independent attention networks achieve parallelization. The multi-head attention layer takes inputs and is fed to three layers to create the query(Q), key(K), and value(V) vectors [30].

$$Attention = Softmax(\frac{QK_T}{\sqrt{dK}})V$$

The combination of the n attention layer gives a multi-head attention layer.

$$Multihead\ attention(Q, K, V) = Concat(head_1, \dots \dots \dots .. head_n)$$

wherehead$_i$ = $Attention\left(QW_i^Q,\ KW_i^K, VW_i^V\right)$

Like the encoder block, the decoder has positional encoders and masked multi-head attention layers. The attention vectors from the encoder block and its masked multi-head attention layers are handed to another multi-head attention block. The feed-forward network is passed with a set of vectors where each vector gives the relation with other words in the entire document. Finally, a Softmax function is used after the linear layer is placed after the feed-forward network [29] at the output to convert it into a probability distribution function.

Figure 3 and Figure 4 depict the model architecture of the transformer model that contains encoder and decoder layers connected with feed-forward networks, attention layers, and normalization layers.

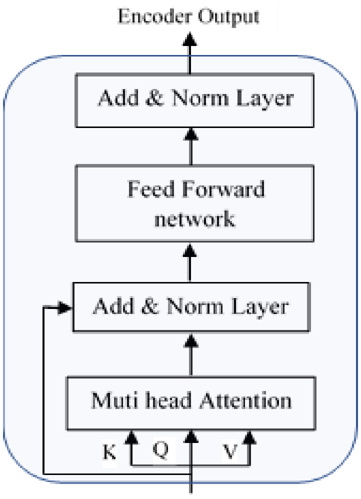


Fig. 3 Transformer Encoder

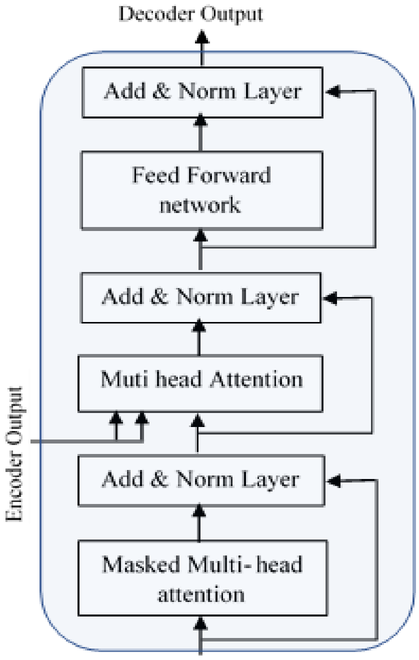


Fig. 4 Transformer Decoder

The proposed transformer model is designed with varying layers of encoder, decoder, and feed-forward network layers and accordingly named with different model numbers as 1, 2, and so on. Finally, six models were formed and tested. The following are the algorithm steps that are used while designing the Transformer.

Transformer Algorithm steps:

1. ***Find the word embedding for the input sequence:***
   - ***Use a pre-trained word embedding model like Word2Vec to convert each word in the input sequence to a vector representation.***
2. ***Find the positional information (Positional encoding):***
   - ***Generate positional encoding vectors to add to the word embeddings.***
   - ***For each position in the input sequence, generate a positional encoding vector using a formula like a sin(pos/10000^(2i/dim)) or cos(pos/10000^(2i/dim)), where pos are the position, and i is the index of the dimension in the embedding vector, anddim is the dimensionality of the embedding vector.***
   - ***Add the positional encoding vectors to the word embeddings.***
3. ***Summing of word embedding and positional encoding:***
   - ***Add the word embeddings and positional encoding vectors element-wise to get the final embeddings for each word in the input sequence.***

4. *Apply the query (Q), key (K), and value (V) vectors to the Transformer Encoder:*
    - *Feed the embeddings to a Transformer Encoder layer.*
    - *The encoder generates Q, K, and V vectors for each position in the input sequence.*
    - *Use these vectors to calculate attention scores between all positions in the series.*
    - *Use these scores to weight the values (V) vectors and generate a context vector for each position in the sequence.*
    - *Feed the context vectors to the next Transformer Encoder layer.*
5. *Repeat the Transformer Encoders for N number of times:*
    - *Repeat step 4 for N number of Transformer Encoder layers.*
6. *Find the word embedding for the ground truth sequence:*
    - *Convert each word in the ground truth sequence to a vector representation using the same pre-trained word embedding model as in step 1.*
7. *Find the positional information (Positional encoding):*
    - *Generate positional encoding vectors for the ground truth sequence using the formula in step 2.*
    - *Add the positional encoding vectors to the word embeddings.*
8. *Summing of word embedding and positional encoding and applied to the Transformer Decoder:*
    - *Add the word embeddings and positional encoding vectors element-wise to get the final embeddings for each word in the ground truth sequence.*
    - *Feed the embeddings to a Transformer Decoder layer.*
    - *The decoder generates Q, K, and V vectors for each position in the ground truth sequence.*
    - *Use the Q vectors and attention scores calculated from the encoder to generate context vectors for each position in the ground truth sequence.*
    - *Feed the context vectors to the next Transformer Decoder layer.*
9. *The decoder also receives the output of the encoder, which now allows the decoder to attend to all the words in the input sequence:*
    - *The context vectors generated in step 8 are used to calculate attention scores between all positions in the input sequence.*
    - *Use these scores to weight the values (V) vectors and generate a context vector for each position in the input sequence.*
    - *Feed the context vectors to the next Transformer Decoder layer.*
10. *Repeat Transformer Decoders for N number of times:*
    - *Repeat steps 8 and 9 for N number of Transformer Decoder layers.*
11. *The output of the decoder finally passes through a fully connected layer:*
    - *Feed the final context vectors generated by the last Transformer Decoder layer to a fully connected layer.*
12. *Followed by a Softmax layer, generate a prediction for the next word of the output sequence:*
    - *Apply a softmax activation function to the output of the fully connected layer to get the probability distribution over the vocabulary.*
    - *Select the word with the highest probability as the prediction for the next term of the output sequence.*

Initially, basic transformer model 1 is implemented with four layers of encoder and decoder, and four feed-forward network layers are considered. Later, the number of encoder, decoder, and feed-forward network layers varies in different models for better performance. The models implemented are represented in Table 1. For instance, model_6 is implemented with three encoder and decoder layers, and the number of feed-forward network layers is 4.

Table 1 Different transformer models with layer details

Model	Encoder layers	Decoder layers	No. of layers in feed-forward network
Model_1	4	4	4
Model_2	5	5	4
Model_3	3	3	2
Model_4	4	4	2
Model_5	5	5	2
Model_6	3	3	4

The Model_6 outperforms other models for the following reasons:

- Model_6 has fewer encoder and decoder layers (3) compared to model_1, model_2, model_4, and model_5, which have 4 or 5 layers. This simplicity results in faster performance, potentially improving efficiency. While a more complex model might capture more abstract patterns, it could also demand more computation and memory or risk overfitting the data or missing essential information.
- The feed-forward network in model_6 boasts more layers (4) than model_3, model_4, and model_5, which only have two layers. This makes model_6 broader and more versatile, enhancing its ability and adaptability. A more general model introduces more variables and possibilities but raises the risk of overfitting.
- Model_6 may be more suitable for the task and data, balancing depth and width. The optimal number of layers depends on factors such as the data's type and size, the task's goal and challenges, and the model's complexity and efficiency.
- Model_6 also has more optimal hyperparameters, such as the number of attention heads, hidden size, dropout rate, learning rate, etc., compared to other models. These hyperparameters significantly influence the training and evaluation of the model, necessitating careful selection and adjustment to align with the task and data requirements

3.3. RNN Model

Recurrent Neural Network (RNN) [31] is an artificial neural network designed for sequence data, making it well-suited for several tasks, including natural language processing, etc. The critical feature of capturing and remembering information from earlier inputs in the sequence makes it an effective tool when context or temporal dependencies are essential. The data is processed sequentially in RNN [32, 33,35], as shown in Figure 5. Once the first word of a sentence is processed, a hidden state is generated and is given as input to the encoder along with the next word and so on, and at last, it is given to the decoder. Long-term dependency becomes a problem in the case of long sentences due to the vanishing gradient problem. It is inefficient and time-consuming because of its sequential nature of training and difficulty with parallel training. Also, it needs help to efficiently capture and model temporal lags in the data, limiting its ability to make accurate predictions where time-dependent patterns are involved.

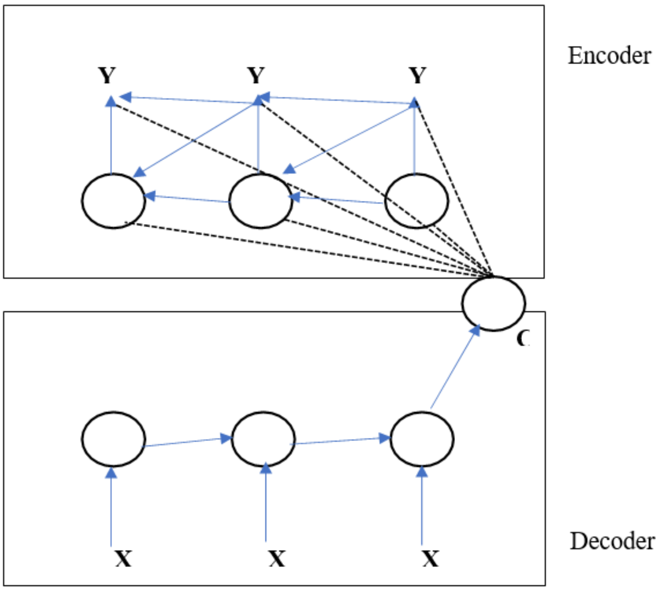


Fig. 5 Recurrent neural network Architecture

The encoder-decoder architecture has been foundational for sequence-to-sequence tasks such as machine translation, text summarization, etc. In the RNN encoder-decoder architecture shown in Figure 5, a variable-length sequence is encoded into a fixed-length vector representation and decoded into another variable-length sequence. The encoder is an RNN that reads each symbol of an input sequence sequentially. Each symbol is read, gives a hidden state, and is given to the next stage, and finally, the summary, C, corresponding to the whole input sequence, comes out. This is taken by the decoder, trained to generate the output sequence by predicting each word corresponding to each stage.

4. Results and Discussion

In this paper, six variations of deep learning-based Transformers are considered, in addition to RNN, for training and testing to generate text summarization of the Inshorts database. The short input articles shown in Table 2 are Inshorts database test articles. The 'ground truth headline' (summary) represents the actual headline given by an expert. The 'predicted headline' represents the summary predicted by the pre-trained model when the input article is provided. The ground truth and predicted summaries are then compared. If they are similar, the model is considered to have predicted the summary correctly, resulting in high ROUGE metrics; otherwise, its performance is deemed low.

Table 2 Random samples of in-short database summarization using model_6

S.No	Short	Predicted headline	Ground truth headline
Article 1	As per a survey of Britons by Scottish investment company Scottish Widows, former British PM Margaret Thatcher is the most influential Woman of the last two centuries. With 28% votes, Thatcher was ranked ahead of Marie Curie (24%) and Queen Elizabeth II (18%). Princess Diana, Mother Teresa, and Oprah Winfrey made it to the top ten of the list.	Thatcher voted most influential Woman: UK poll.	Margaret Thatcher, most influential Woman of the last two centuries: Survey
Article 2	On Wednesday, Tata Motors posted a 16% year-on-year rise in its global sales to 93,355 units for January. While sales of passenger vehicles increased by 14% to 56,616 units, the commercial vehicles segment recorded a 20% growth to 36,739 units. Further, sales of luxury brand Jaguar Land Rover jumped by 25% to 45,535 units.	Tata Motors global sales up by 16% in January	India's Tata Motors posted a 16% rise in its global sales for January
Article 3	Jet Airways today confirmed that the five crew members who allowed playback singer Sonu Nigam to sing onboard the Jaipur-Mumbai flight on January 4 using the announcement system have been &#34; taken off flight duty&#34; A statement from the airline added that an inquiry had been ordered to ensure strict adherence to the flying norms in the future.	Jet crew suspended for Sonu Nigam& #39;s song on board	Jet Airways confirms that crew members w ho allowed Sonu Nigam to sign off the flight have been taken off flight duty

The RNN and the proposed transformer-based models are evaluated on an in-shorts dataset [22]. In line with this, full-length F1 scores of ROUGE-1 and ROUGE-L metrics are calculated. ROUGE is an acronym for Recall-Oriented Understudy for Gisting Evaluation, a method used to measure the quality of summaries (predicted) by comparing them with human-written summaries (ground truth). ROUGE has different variants, including ROUGE-N, ROUGE-L, and ROUGE-S, which calculate the similarity of n-grams, longest common subsequences, and skip-bigrams between the summaries. Various factors, such as the number, order, and meaning of the words in the outlines, influence the ROUGE score. A higher ROUGE score indicates that the summary is closer to the human-written summary, which is typically the standard for quality. However, a higher ROUGE score implies a good summary.

Here, the ROUGE-1 score gives the unigram coincidence between the predicted and ground truth summaries (summary given by expert). In contrast, ROUGE-L gives the longest sequence coincidence between the expected and ground truth summaries. The number of epochs indicates the number of times the entire dataset is experienced to the model, and training loss suggests how close the predicted overview of the model is to the ground truth summary.

Table 3 Comparison of training loss of various models

Epochs	RNN	Model_1	Model_2	Model_3	Model_4	Model_5	Model_6
1	7.0099	10.283	10.2868	10.289	10.2776	10.2783	10.2805
5	4.5356	5.3965	5.6915	5.2428	5.6761	5.3996	4.9721
10	3.7701	4.0926	4.5996	3.6784	4.6188	4.6366	3.2273
15	3.3071	3.6132	4.1459	2.9958	4.1504	4.1672	2.5478
20	2.9695	3.3175	3.916	2.4741	3.9022	3.905	2.0363
25	2.7154	2.5984	3.6958	2.1308	2.8659	3.6793	1.9923
30	2.5125	2.0118	3.5862	1.9091	2.1266	3.4529	1.7213
35	2.5522	1.8615	3.0541	1.7533	1.9691	2.8615	1.5527
40	2.5673	1.6587	2.6953	1.6222	1.7211	2.0514	1.4692
45	2.5688	1.6432	2.6544	1.6160	1.7122	1.9904	1.4456
50	2.5690	1.6422	2.6512	1.6114	1.7119	1.9900	1.4399

The training loss of six different transformer models and one RNN model is shown in Table 3. As the models are trained for more epochs, the loss decreases gradually. Model_6 has the highest loss of 10.2805 at epoch 1, but it reduces to 1.4399 at epoch 50, the lowest among all models. The RNN model starts with a loss of 7.01 and ends with a loss of 2.5690 after 50 epochs. Therefore, model_6 has the best performance with a training loss of 1.4399.

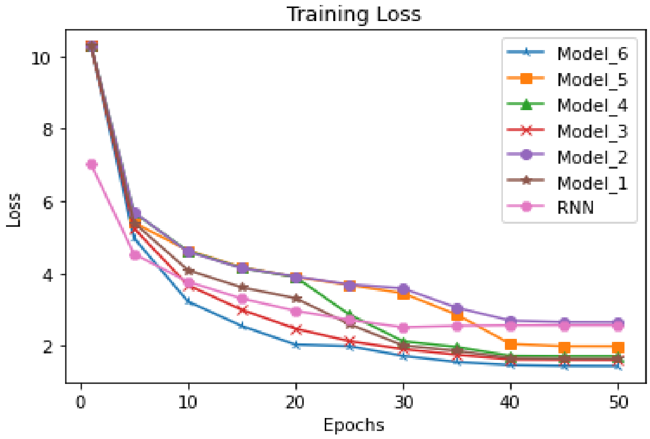


Fig. 6 Comparison of training loss for proposed models

The comparison of training loss between six transformer models and one RNN model is presented in Figure 6. All the models are built from scratch. Model_6 has the lowest loss values among all models, indicating its superior performance. The other models have higher loss values than Model_6.

Table 4 Comparison of training accuracies of various models

Epochs	RNN	Model_1	Model_2	Model_3	Model_4	Model_5	Model_6
1	0.2499	0	0	0	0	0	0
5	0.4364	0.1602	0.1585	0.1717	0.1051	0.172	0.1917
10	0.4679	0.2093	0.2094	0.2387	0.1809	0.2228	0.2983
15	0.4955	0.2465	0.2476	0.2873	0.2675	0.2575	0.3786
20	0.5195	0.2747	0.276	0.3221	0.3363	0.2888	0.4379
25	0.54	0.3122	0.2987	0.3487	0.3902	0.3131	0.4812
30	0.559	0.3855	0.3677	0.4169	0.4335	0.3303	0.5146
35	0.562	0.4347	0.4122	0.4487	0.4693	0.4122	0.5913
40	0.566	0.5422	0.4485	0.5869	0.679	0.4978	0.7285
45	0.569	0.5587	0.4587	0.5905	0.6872	0.5011	0.7312
50	0.571	0.5590	0.4590	0.5911	0.6890	0.5031	0.7354

The training accuracy of six transformer models and one RNN model is shown in Table 4. The accuracy increases gradually as the models are trained for more epochs. Model_6 has the lowest accuracy of zero at epoch 1, but it improves to 73.54% at epoch 50, the highest among all models. The RNN model has an accuracy of 24.99% at epoch 1 and 57.5% at epoch 50. Therefore, model_6 performs best with a training accuracy of 73.54%, while the RNN model is inferior.

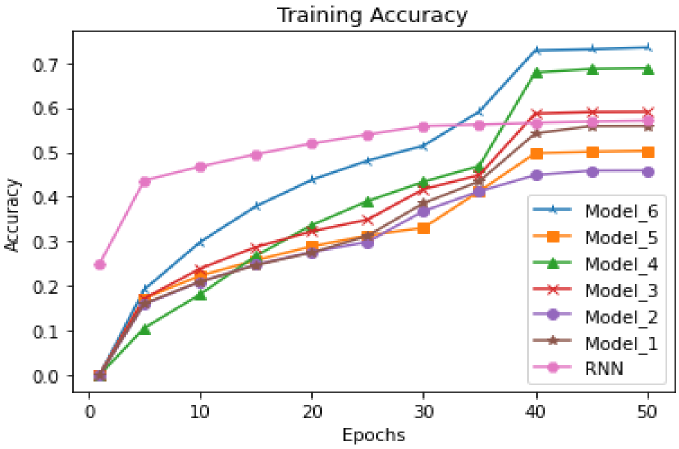


Fig. 7 Comparison of training accuracy for proposed models

The training accuracies of six transformer models and one RNN model are compared in Figure 7. All the models are built from scratch. Model_6 has the highest training accuracy of over 70%, while the other models have lower accuracies than it.

Table 5 Comparison of ROUGE metrics of various models

Model	Rouge-1	Rouge-L
RNN	0.175	0.134
Model 1	0.2277	0.2275
Model 2	0.111	0.11
Model 3	0.318	0.266
Model 4	0.232	0.232
Model 5	0.2235	0.2185
Model 6	**0.36**	**0.3305**

The rouge metrics of six transformer models and one RNN model are shown in Table 5. ROUGE metrics measure the quality of the summary generated by the model by comparing it with the reference summary. A higher ROUGE value indicates a more efficient model. Model_6 has the highest ROUGE-1 value of 0.36 and ROUGE-L value of 0.3305, making it the most efficient among all models. Therefore, the machine's performance using the RNN model can be improved using a transformer-based model instead.

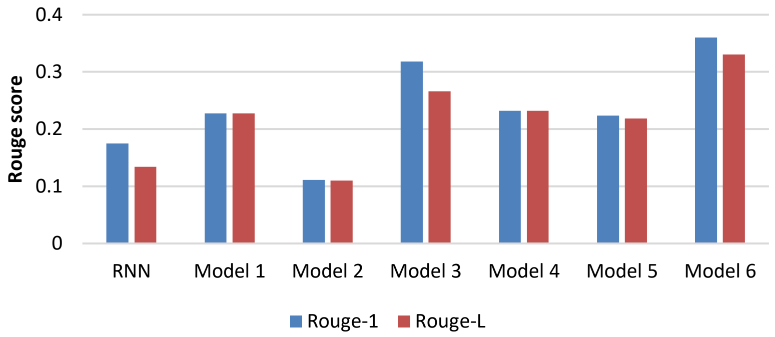


Fig. 8 Comparison of Rouge metrics of various models

The comparison of Rouge-1 and Rouge-L scores between six transformer models and one RNN model is presented in Figure 8. Model_6 performs best with the highest values for both Rouge-1 and Rouge-L scores. The other models have lower scores than Model_6.

The proposed transformer model, Model _6, is also compared with existing abstractive text summarization models such as PEGASUS-CNN_dailymail, BART-large-CNN, and DistilBART-CNN-12-6 with respect to ROUGE-1, ROUGE-L metrics as shown in Table 6. The large-sized BART (BARTlarge) model is fine-tuned on CNN/Dailymail to get the BART-large-CNN model. It has 12 layers each for encoder, decoder, and hidden state. PEGASUS-CNN_dailymail is obtained by fine-tuning the PEGASUS-large model on CNN/Dailymail. DistilBART-CNN-12-6 is a smaller version of BART trained on CNN/Dailymail. It has 12 layers for the encoder and hidden state but only 6 for the decoder.The proposed model_6 is implemented with three encoder and decoder layers and several feed-forward network layers 4. It is trained on the CNN/Dailymail news dataset. CNN/Dailymail [34] dataset contains over 300,000 news articles from CNN and the Daily Mail newspaper, written between 2007 and 2015. Each article is accompanied by a list of bullet point summaries that abstractively summarize the headline.

Table 6 Comparison of ROUGE metrics of the proposed model with existing models

Model	Rouge-1	Rouge-L
PEGASUS-CNN_dailymail[2]	0.4186	0.2025
BART-large-CNN [2]	0.4270	0.2058
DistilBART-CNN-12-6 [2]	0.4292	0.2077
Proposed Model	**0.4297**	**0.2103**

From Table 6, the proposed model_6 has the highest scores on ROUGE-1 and ROUGE-L metrics, measuring the overlap between the generated and reference summaries. This means that the proposed model_6 is more effective and accurate than the other models in producing abstractive summaries.

5. CONCLUSIONS

The six proposed transformer-based models, each featuring varying numbers of encoders, decoders, and feed-forward network layers, reveal superior performance compared to the RNN model. The metrics such as training loss, training accuracy, and ROUGE scores are compared with the baseline RNN model. Among these transformer models, results show that model_6 has the best performance with three encoder and decoder layers and four feed-forward network layers. Model_6 has the lowest training loss, highest training accuracy, and highest ROUGE-1 and ROUGE-L scores. Model_6 is compared with existing abstractive text summarization models such as PEGASUS-CNN_dailymail, BART-large-CNN, and Distil BART-CNN-12-6 on the CNN/Dailymail dataset. The superiority of model_6 is analyzed by considering its simplicity, versatility, balance, and optimality. This unequivocally establishes model_6 as the most effective transformer model under consideration. These models can be applied to other natural language generation tasks, such as machine translation, paraphrasing, and question-and-answer generation. In future work, more advanced technologies and methods involving explainable AI to explore these tasks in generating summaries can be considered with user preferences. Also, the proposed models can be applied to datasets in other languages.

REFERENCES

[1] M. Alfraheed, "An Approach for Features Matching between Bilateral Images of Stereo Vision System Applied for Automated Heterogeneous Platoon", *Journal of Theoretical & Applied Information Technology,* vol. 96, no.7, Apr. 2018.

[2] N. Giarelis, M. Charalampos, and K. Nikos, "Abstractive vs. Extractive Summarization: An Experimental Review", *Applied Sciences*, vol. 13, no. 13, Jun 2023.

[3] Y. Chen, M. Yun, M. Xudong, and L. Qing, "Multi-task learning for abstractive and extractive summarization", *Data Science and Engineering*, vol. 4, pp. 14–23, Mar 2019.

[4] D. Suleiman, and A. Arafat, "Deep learning based abstractive text summarization: approaches, datasets, evaluation measures, and challenges", *Mathematical problems in engineering,* pp. 1–29, Aug 2020.

[5] R. S. Shini, and K. VD. Ambeth, "Recurrent neural network based text summarization techniques by word sequence generation", In Proceedings of the 6th International Conference on Inventive Computation Technologies (ICICT), Jan. 2021, pp. 1224–1229.

[6] W. Fang, J. TianXiao, J. Ke, Z. Feihong, D. Yewen, and S. Jack, "A method of automatic text summarization based on long short-term memory", *International Journal of Computational Science and Engineering*, vol. 22, no. 1, pp. 39–49, 2020.

[7] T. Goyal, J. L. Junyi, and D. Greg, "News summarization and evaluation in the era of gpt-3." arXiv preprint arXiv: 2209.12356, Sep. 2022.

[8] W. S. El-Kassas, R. S. Cherif, A. R. Ahmed, and K. M. Hoda, "Automatic text summarization: A comprehensive survey", *Expert systems with applications*, vol. 165, Mar 2021.

[9] K. Ganesan, XZ. Cheng, and H. Jiawei, "Opinosis: A graph-based approach to abstractive summarization of highly redundant opinions." In Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), 2010, pp. 340-348.

[10] P.-E. Genest, and L. Guy, "Fully abstractive approach to guided summarization." In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, 2012, vol. 2, pp. 354–358.

[11] A. Khan, S. Naomie, F. Haleem, K. Murad, J. Bilal, A. Awais, A. Imran, and P. Anand, "Abstractive text summarization based on improved semantic graph approach", *International Journal of Parallel Programming*, vol. 46, pp. 992–1016, 2018.

[12] T. Shi, K. Yaser, R. K. Naren, and K. R. Chandan, "Neural abstractive text summarization with sequence-to-sequence models", *ACM Transactions on Data Science*, vol. 2, no. 1, pp. 1–37, 2021.

[13] X. Chen, G. Shen, T. Chongyang, S. Yan, Z. Dongyan, and Y. Rui, "Iterative document representation learning towards summarization with polishing." arXiv preprint arXiv: 1809.10324, 2018.

[14] L. Dong, Y. Nan, W. Wenhui, W. Furu, L. Xiaodong, W. Yu, G. Jianfeng, Z. Ming, and H. Hsiao-Wuen, "Unified language model pre-training for natural language understanding and generation", *Advances in neural information processing systems*, vol. 32, 2019.

[15] X. Zhang, and L. Mirella, "Sentence simplification with deep reinforcement learning", arXiv preprint arXiv: 1703.10931, 2017.

[16] J. Lin, S. Xu, M. Shuming, and S. Qi, "Global encoding for abstractive summarization", arXiv preprint arXiv: 1805.0398, 2018.

[17] R. Pasunuru, and B. Mohit, "Multi-reward reinforced summarization with saliency and entailment", arXiv preprint arXiv: 1804.06451, 2018.

[18] N. Kalchbrenner, E. Lasse, S. Karen, van den O. Aaron, G. Alex, and K. Koray, "Neural machine translation in linear time", arXiv preprint arXiv: 1610.10099, 2016.

[19] Van den O. Aaron, D. Sander, Z. Heiga, S Karen, V. Oriol, G. Alex, K. Nal, S. Andrew, and K. Koray, "Wavenet: A generative model for raw audio", arXiv preprint arXiv: 1609.03499, vol. 12, 2016.

[20] A. Vaswani, S. Noam, P. Niki, U. Jakob, J. Llion, N. G. Aidan, K. Łukasz, and P. Illia, "Attention is all you need", *Advances in neural information processing systems*, vol. 30, 2017.

[21] D. Bahdanau, C. Kyunghyun, and B. Yoshua, "Neural machine translation by jointly learning to align and translate", arXiv preprint arXiv: 1409.0473, 2014.

[22] M. Lewis, L. Yinhan, G. Naman, G. Marjan, M. Abdelrahman, L. Omer, S. Ves, and Z. Luke, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension", arXiv preprint arXiv: 1910.13461, 2019.

[23] https://www.kaggle.com/datasets/shashichander009/inshorts-news-data.

[24] S. Abbes, B. A. Sarra, H. Rim, and C. Philippe, "Automatic text summarization using transformers", In Proceedings of the Knowledge Graphs and Semantic Web: Third Iberoamerican Conference and Second Indo-American Conference, November 2021, vol. 3, pp. 308–320.

[25] S. Gupta, and K. G. Sanjai, "Abstractive summarization: An overview of the state of the art", *Expert Systems with Applications*, vol. 121, pp. 49–65, May 2019.

[26] C.-Y. Lin, and F. J. Och, "Looking for a few good metrics: ROUGE and its evaluation", in NTCIR workshop, 2004.

[27] T. Wolf, D. Lysandre, S. Victor, C. Julien, D. Clement, M. Anthony, C. Pierric et al, "Hugging face's transformers: State-of-the-art natural language processing", arXiv preprint arXiv:1910.03771, 2019.

[28] C. Raffel, S. Noam, R. Adam, L. Katherine, N. Sharan, M. Michael, Z. Yanqi, L. Wei, and J. L. Peter, "Exploring the limits of transfer learning with a unified text-to-text transformer", *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.

[29] F. Zhuang, Q. Zhiyuan, D. Keyu, X. Dongbo, Z. Yongchun, Z. Hengshu, X. Hui, and H. Qing, "A comprehensive survey on transfer learning", In Proceedings of the IEEE, 2020, vol. 109, no. 1, pp. 43–76.

[30] B. Srinivas, and S. R. Gottapu, "Segmentation of multi-modal MRI brain tumor sub-regions using deep learning", *Journal of Electrical Engineering & Technology*, vol. 15, no. 4, pp.1899–1909, Jul 2020.

[31] M. Allahyari, P. Seyedamin, A. Mehdi, S. Saeid, D. T. Elizabeth, B.G. Juan, and K. Krys, "Text summarization techniques: a brief survey", arXiv preprint arXiv: 1707.02268, 2017.

[32] C. Khatri, S. Gyanit, and P. Nish, "Abstractive and extractive text summarization using document context vector and recurrent neural networks", arXiv preprint arXiv: 1807.08000, 2018.

[33] M. Ranzato, C. Sumit, A. Michael, and Z. Wojciech, "Sequence level training with recurrent neural networks", arXiv preprint arXiv: 1511.06732, 2015.

[34] https://www.kaggle.com/datasets/gowrishankarp/newspaper-text-summarization-cnn-dailymail.

[35] Y. Yu, S. Xiaosheng, H. Changhua, and Z. Jianxun, "A review of recurrent neural networks: LSTM cells and network architectures", *Neural computation*, vol. 31, no. 7, pp. 1235–1270, Jul 2019.