National College of Ireland

Developing Bengali Text Summarization with Transformer Base model

MSc Research Project
Data Analytics

Aditya Mukherjee
Student ID: X20161131

School of Computing
National College of Ireland

Supervisor: Dr. Rejawanul Haque

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Aditya Mukherjee
Student ID:	X20161131
Programme:	Data Analytics
Year:	2021
Module:	MSc Research Project
Supervisor:	Dr. Rejawanul Haque
Submission Due Date:	31/01/2022
Project Title:	Developing Bengali Text Summarization with Transformer Base model
Word Count:	5702
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Aditya Mukherjee	
Date:	30th January 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Developing Bengali Text Summarization with Transformer Base model

Aditya Mukherjee
X20161131

Abstract

For high-resource languages like English and other European languages, text summarization using deep learning has become a well-studied research subject. However, for poorly available resource languages such as Indian subcontinent languages and African languages, relatively few efforts have been done on the Internet. Due to a lack of a sufficient parallel corpus, parser, tokenizer, POS taggers, and other tools, resource-constrained languages have a restricted reach in natural language processing (NLP). We propose an abstractive text summarization sequence for a deep learning model for Bengali in this Research study. This Research study adopt a novel approach towards Summarizing the Bengali text which have been collected from Bangla news corpus and try to implement the LSTM-RNN based Encoder and decoder model with attention mechanism and Transformer based model called multilingual- T5 model and will Evaluate their result with each other using ROUGE metric. In this study we will be using many NLP tools too, to process the data and will clean it before inputting the data into our model.

1 Introduction

We live in a fast-paced world where we are more reliant on mobile and other technologies to make our lives easier. Amount of textual data in the world is growing by the second, and researchers and data scientists all around the world are attempting to find out how to extract meaningful information from the data without having to read the full data set. To solve this problem researchers have come up with Automatic document summarization. A document summarization goal to provide the most important and relevant information in a form of text and present the text in a condensed form.

Text summarizing may be viewed as a difficult task since writing a brief, exact, and fluent summary of a longer text content is difficult. Automatic text summarizing methods are badly needed to handle the ever-increasing volume of text data available online in order to find relevant information and consume relevant information faster. If we think of internet, which consists of web pages, news items, status updates, blogs, and a variety of other things. Because the data is unstructured, the most we can do is conduct a search and skim the results. There is great need of an algorithm or machine learning model which can reduce the length of text, focused summaries that capture every relevant detail of the textual data that we are feeding it. Automatic text summarization is of two types-:

1) Extractive Text Summarization-: Extractive summarization involve, to create the new summary, phrases and sentences that are chosen from the original text. The relevance of phrases is rated in order to select just those that are most important to the meaning of the source.
2) Abstractive Text Summarization-: Abstractive text summarization is process where it create an entire new words and phrases in order to convey the meaning of the original information. This is a more challenging method, but it is the one that humans will eventually choose. Traditional approaches are used to select and compress the material of the source document.

Many of text summarization models are being applied and tested on different type of textual data as well as different language like Hindi, Spanish, German, Russian, Bengali etc. The language of textual data matter to the model and its algorithm. There are only handful of models which are being implemented on Bengali language.

Bengali is one of the widely spoken language in the world and native language of the Bengali community. Bengali is second most spoken language in India out of 22 language and its national language of Bangladesh and over 98% of Bangladesh population speak Bengali as their primary language. Over the span of more than 1,300 years, Bengali has evolved. Bengali literature is one of Asia's most prolific and diverse literary traditions, with a millennium-old literary history that thrived during the Bengali Renaissance. As Bengali literature grow, the amount of textual data has increased. In this era of digitization of media, people don't have time to read whole book, articles, or blogs. On internet there are so many Bengali article which are quite big and need to be reduced to make it more readable. So, building a good Bengali Text summarization method is much needed to get a better perspective and knowledge out of long textual data. And building Bengali text summarization model is difficult as there are not many openly accessible resources available.

1.1 Research Motivation and Background

There are many research going on the field of Bengali text summarization and many of Bengali summarization have been built like in the work by Uddin and Khan (2007) presented a method for summarizing Bengali documents based on extraction. The sentences were ranked using a variety of factors, including geography, term frequency, numerical data, and so on. They created the Bengali summarizer based on the attributes and determined that the summary size should be 40% of the real text. Das and Bandyopadhyay [3] used emotion information to summarize Bengali documents. They attempted to extract sentiment information from a document and then aggregated it to create a summary. According to Sarkar (2012) Text summarizing entails pre-processing, stemming, sentence rating, and summary generation. Stopwords must be removed, stemming must be performed, and the input must be converted into a collection of sentences. These are few of the works which have done in the Bengali Text summarization. Mihalcea Mihalcea and Tarau (2004) Mihalcea (2004) has worked on graph-based text summarization. The sentences can be represented as nodes in a graph, with edges linking them. Edge weights may then be calculated by calculating the similarity of two nodes.

Till now most used model to summarize Bengali text is LSTM based Encoder and decoder model. Until now, one of the greatest approaches to capture the timely dependencies in

sequences was to use recurrent networks. However, the authors of the article Joshi et al. (2019) demonstrated that an architecture based only on attention processes that is transformer models, rather than RNNs (Recurrent Neural Networks), can enhance performance in machine translation and other activities.

Therefore, the domain needs a transformer model that can enhance the performance of the Bengali text summarization model and give accurate text summarization. Transformer model can find pattern within the data which helps them to give the result more efficiently than the RNN model. Transformer model mechanism works by taking data, encoding it, and then recording how each word connects to the words that come before and after it. Here we will be using MT5 transformer model for the Bengali Abstractive text summarization. The working of transformer model is shown in figure[1]

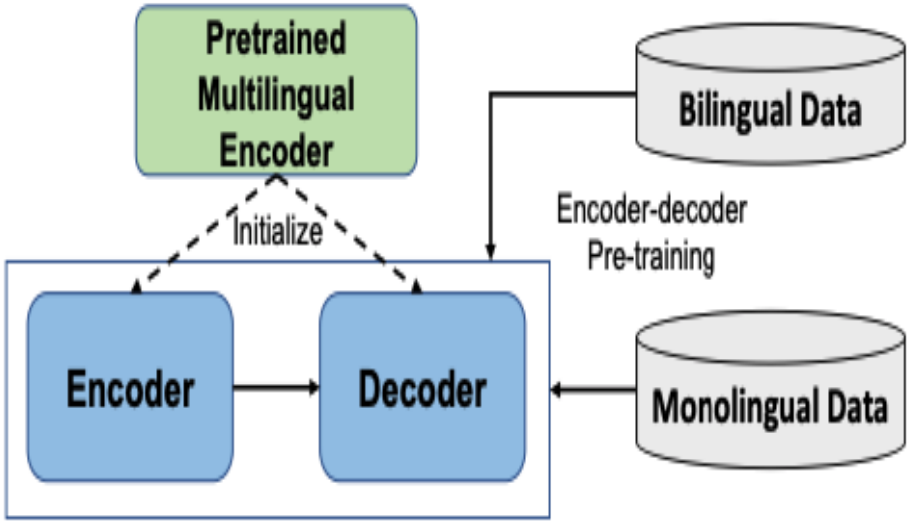

Figure 1: Transformer model[1]

1.2 Research Question

How well can Fine-tuned Multi-lingual T5 model can perform in comparisons to LSTM based encoder-decoder model?

1.3 Research Pattern

Table 1: Research Pattern

Objective	Description	Metrics
1	A critical review of existing method	
2	Data pre-processing and EDA	
3	Implementation of LSTM model	
4	Implementation of MT5 model	
5	Comparison of MT5 and LSTM model	Rouge metric and Summaries

2 Related Work

This section provides an overview of many types of research relevant to automated text summarization. The majority of text summarizing research is centered on English materials. Despite the fact that Bengali is the world's seventh most spoken language, there have been few studies on automatic Bengali text summarization.

2.1 Tranformer based Model Implementation

While the architecture of all of these models is almost identical, the self-supervised tasks performed in the pretraining stage varies significantly. These objectives target the models to learn broad elements of the language, such as masking tokens or rearranging sentences, in certain situations, such as BART, T5, and ProphetNet. BART has been pre-trained to rebuild masked spans (text infilling) and to reassemble phrases after they have been permuted (sentence permutation). T5 is also pre-trained on encoder-decoder masked language modeling in order to solve all text-based language difficulties in a text-to-text format globally. Pretraining ProphetNet on future n-gram prediction encourages the model to prepare for future tokens rather than the next token, preventing overfitting on strong local correlations. In other situations, such as PEGASUS, the self-supervised tasks are designed to seem like the summation task in order to foster whole-document comprehension and summary-like creation. Unlike prior models, PEGASUS is trained via Gap Sentences Generation (GSG), which entails rebuilding sentences that maximize the ROUGE over the whole document. According to the authors of PEGASUS, GSG is more suited to abstractive summarization than other pretraining methodologies since it closely reflects the downstream job.
News Abstractive Summarizing for Catalan is a monolingual news summarization methodology developed for the paper (NASca). BART Lewis et al. (2019) is a Transformer encoder-decoder model with the same architecture and hyper-parameters. They opted to integrate different pretraining tasks to introduce linguistic information into the pretraining stage with the goal of boosting the abstractivity of the summaries generated by the model, inspired by the work of Cohn et al. (2020). Sentence permutation, text infilling [6, Gap Sentence Generation (GSG) Zhang et al. (2020), and Next Segment Generation (NSG) [27] were the four tasks that were integrated. NASca is pre-trained simultaneously with the four tasks, which are chosen at random from a uniform distribution in each batch.

Abstractive summarizing research has traditionally concentrated on the development of models employing methods other than those employed in extractive summarizationZhong et al. (2020) Inui et al. (2019)Nallapati et al. (2017)Rush et al. (2015)Nallapati et al. (2016)See et al. (2017). Abstractive summarizers have recently been popular owing to their tremendous generation capabilities, which are attained by pretraining them with self-supervised language modeling tasks on large text corpora and employing encoder-decoder architectures with Transformers Vaswani et al. (2017) as the backbone. The state of the art in abstractive summarization benchmarks is PEGASUS Zhang et al. (2020), BART Lewis et al. (2019), T5 Raffel et al. (2019), and ProphetNet Qi et al. (2020), which are fine-tuned for summarizing tasks.

All of the models and recommendations presented in this section are aimed towards the English language, however there are many more languages that need to be con-

sidered. Multilingual models such as mBART Liu et al. (2020) and mT5 Xue et al. (2020) have been used to consider additional languages alongside the English language. Although these efforts are practical and effective in many circumstances, the performance of multilingual models is often poorer on languages that are underrepresented in the pretraining data or deviate significantly from the most represented languages in terms of linguistic keywords Virtanen et al. (2019)Pires et al. (2019). Pretraining monolingual BERT models were used to investigate learning monolingual models from scratch for language understanding, with outstanding results in various languages such as French Martin et al. (2019)Le et al. (2019) Dutch de Vries et al. (2019), and Spanish Canete et al. (2020)Gonzalez et al. (2021).

2.2 Bengali Text summarization Approaches

Sarkar (2012) has studied automatic text summarization for only one single document which is in Bengali dialect and highlight the influence of position feature of sentences and thematic term feature. Thematic feature is a linguistic term that refers to how something relates to the topic of a piece of literature. Preprocessing, sentence rating, and summary production were the three primary phases of the project. Thematic words and sentence position were used to rate sentences. In their earlier work Sarkar (2014), they have described a key-based approach to summarizing that focused on extracting a collection of key words from a document and constructing an extractive summary based on them. Single-word or multi-word key phrases are acceptable. He utilized two separate datasets, one for English and one for Bengali. In compared to prior research, he judged that the findings were pretty satisfactory. Srivastava and Gupta Srivastava and Gupta (2014) used Extract Technology to provide a summary based on the frequency of terms in their study. They proposed the Gradual NLP algorithm, which is an NLP (Natural Language Processing)-based approach. Analysis, development, and synthesis are the three steps of the summation process. The analyzation step examines the data's text and picks a few essential qualities. The transformation procedure converts empirical data into a graphical representation. The synthe- sis process next takes the summary representation and generates an appropriate summary that meets the user's requirements. The algorithm estimates the average frequency after counting the overall frequency of words other than stopwords. The frequency of sentences with cue words included in them is enhanced for summary generation, and sentences with a score larger than the average frequency are picked for a summary.
Chandro et al. (2018) experimented with extraction-based summarization strategies by collaborating individual words and scoring phrases. Experimentation documents were gathered from famous Bengali daily newspapers. They ranked sentences based on Term Frequency, Positional Value, Connecting Words, and Document Sentence Length. Using these factors, sentences were sorted, and the top K ranked sentences were chosen for the summary. Uddin and Khan (2007) studied Bengali text summarization and found that sentence placement, cue phrase presence, title word presence, term frequency, and numerical data were all significant. They claim that sentences that come at the beginning or end of a passage are more important. Furthermore, the presence of trigger phrases, terms from titles, high-frequency words, and numerical data adds weight to a statement.

Efat et al. (2013) investigated Bengali summary while taking a variety of criteria into account. They graded sentences based on frequency, sentence placement, trigger words,

and other characteristics. After generating scores based on a variety of factors, the final sentence scores were calculated as a weighted sum of the individual feature ratings. They discovered that 83.57 percent of summary phrases correlated to human-created summaries

The usage of key terms in Bangla summarization was investigated by Haque et al. (2016). They emphasized sentences with numerical figures and ordered the sentences in increasing order based on their scores. After the scores were added together, the sentences were ranked. The dataset was created using 400 newspaper documents of various sorts. They stated that utilizing ROUGE-1 and ROUGE-2 increased the quality of their summaries. Das and Bandyopadhyay (2010) summarized Bengali documents using sentiment information. They used a classification approach based on support vector machines. There are three types of characteristics considered: lexico-syntactic features, syntactic features, and discourse level features. The work includes features such as parts of speech, Senti-WordNet, frequency, stemming, chunk label, de-pendency parsing depth, document title, first paragraph, term distribution, and collocation.

3 Methodology

In this research paper, modified Knowledge Discovery in database (KDD) have been used to meet the objective of this research. In this research paper, a Bengali Text Summarizer has been developed which will generate an abstractive summary out of Bengali documents. The whole process flow diagram of the proposed model is shown in below figure.[2]

3.1 Input Document

Any of the Bengali news document can be used as input and can be feed to summarizer. According to research by several sectors, it is found that consumer or user are more interested in in reading news concerning accidents, entertainment, economy, and politics. For any summary, there is significant amount of data needed. There only few amounts of dataset are available in Bengali newspaper. For this research paper data has been collected by Kaggle. The dataset is only made for Abstractive news summarization purpose only. The authors of the dataset have built a data crawler and crawled the data from bangla.bdnews24.com and fetched 19k article and their summaries out of the website and standardized the data.

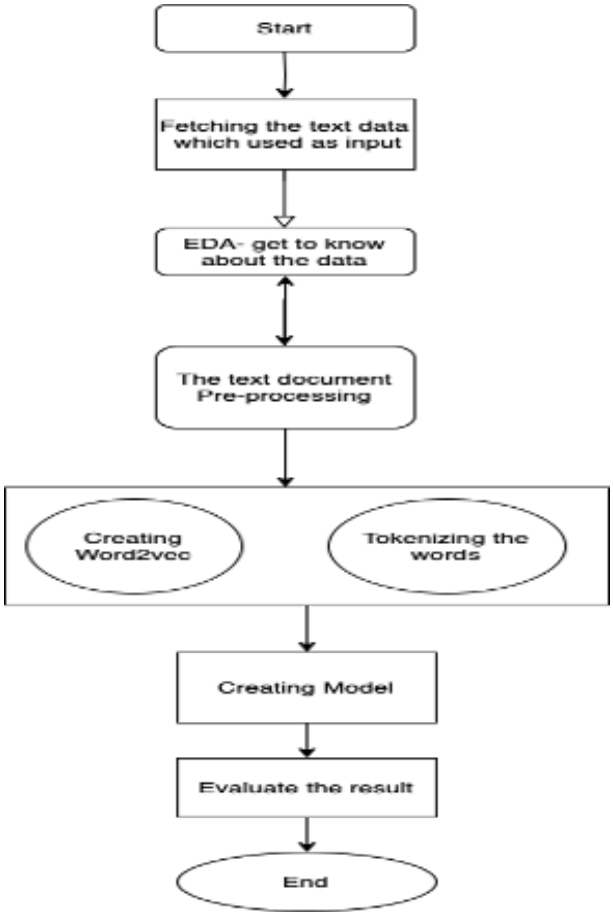

Figure 2: Process flow

3.2 Explanatory data analysis

News: 1
Text: স্টান্ডার্ড চার্টার্ড ব্যাংকের নতুন প্রধান নির্বাহী কর্মকর্তা হিসেবে দায়িত্ব পেয়েছেন আবরার এ আনোয়ার।
Summary: স্টান্ডার্ড চার্টার্ডের নতুন সিইও আবরার
News: 2
Text: রাজধানী থেকে চামড়া শিল্পগুলো সাভারে স্থানান্তরে সিইটিপি ছাড়া সরকার সব কাজ শেষ করেছে বলে জানিয়েছেন শিল্পমন্ত্রী আমির হোসেন আমু।
Summary: মার্চের মধ্যে সাভারে চামড়া শিল্পর সিইটিপি: মন্ত্রী
News: 3
Text: দেশীয় শিল্প বিকাশে সরকারের সব ধরনের উদ্যোগ অব্যাহত রাখার আশ্বাস দিয়েছেন শিল্পমন্ত্রী আমির হোসেন আমু।
Summary: ওয়ালটন কারখানায় শিল্পমন্ত্রী
News: 4
Text: একীভূত হতে চলেছে অনলাইনে শ্রেণিবদ্ধ বিজ্ঞাপন সেবাদাতা দুই প্রতিষ্ঠান এথানেই ডটকম এবং ওএলএক্স।
Summary: একীভূত হচ্ছে এথানেই ডটকমওএলএক্স
News: 5
Text: যাত্রীবাহী একটি বাসে আগুন দেওয়ার আধা ঘণ্টার মধ্যে এই ঘটনায় জড়িত অভিযোগে নড়াইলের পৌর মেয়র ও জেলা বিএনপির সাংগঠনিক সম্পাদক জুলফিকার আলীকে আটক করেছে পুলিশ।
Summary: বাসে আগুন: নড়াইলের পৌর মেয়র গ্রেপ্তার

Figure 3: news vs summary

	article	summary
0	summarize: স্ট্যান্ডার্ড চার্টার্ড ব্যাংক প্রধ...	স্ট্যান্ডার্ড চার্টার্ড সিই আবর
1	summarize: রাজধানী চামড়া শিল্প সাভার স্থানান্ত...	মার্চ সাভার চামড়া শিল্প সিইটিপি: মন্ত্রী
2	summarize: দেশী শিল্প বিকাশ সরকার ধরন উদ্যোগ অ...	ওয়ালটন কারখানা শিল্পমন্ত্রী
3	summarize: একীভূত অনলাইন শ্রেণিবদ্ধ বিজ্ঞাপন স...	একীভূত ডটকমওএলএক্স
4	summarize: যাত্রীবাহী বাস আগুন দেওয় আধা ঘণ্ট ঘ...	বাস আগুন: নড়াইল পৌর মেয়র গ্রেপ্ত

Figure 4: Clean summary

In above figure's[3][4], the article and its respective summary is displayed. All the text have been cleaned and preprocessed.

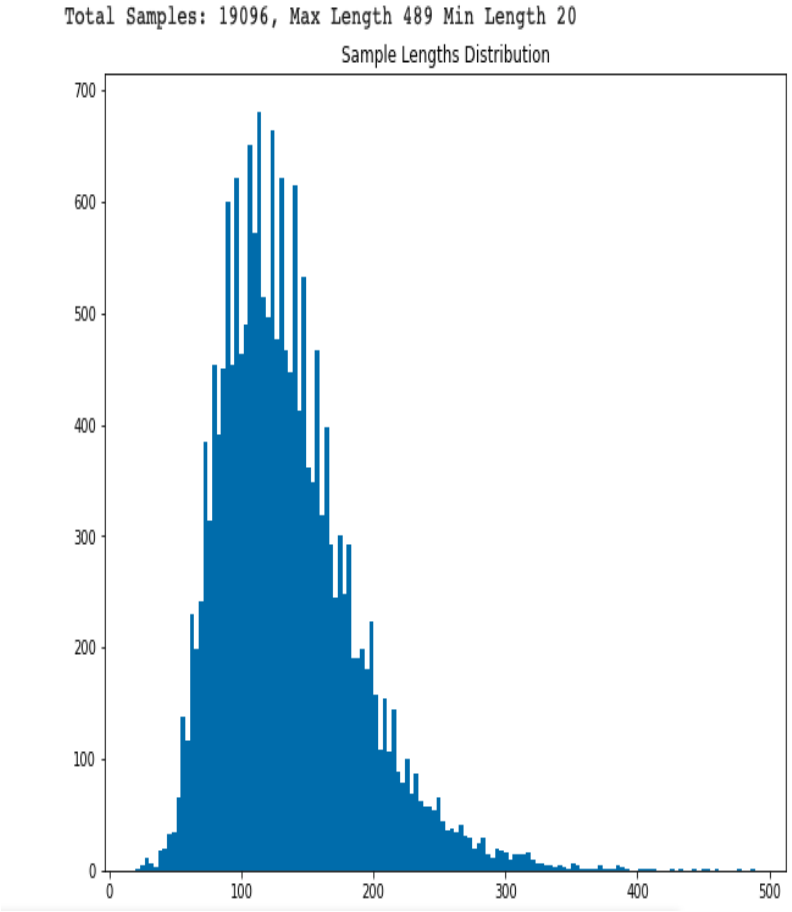

Figure 5: Length of the articles

In above figure[5] the length of the article has been plotted.

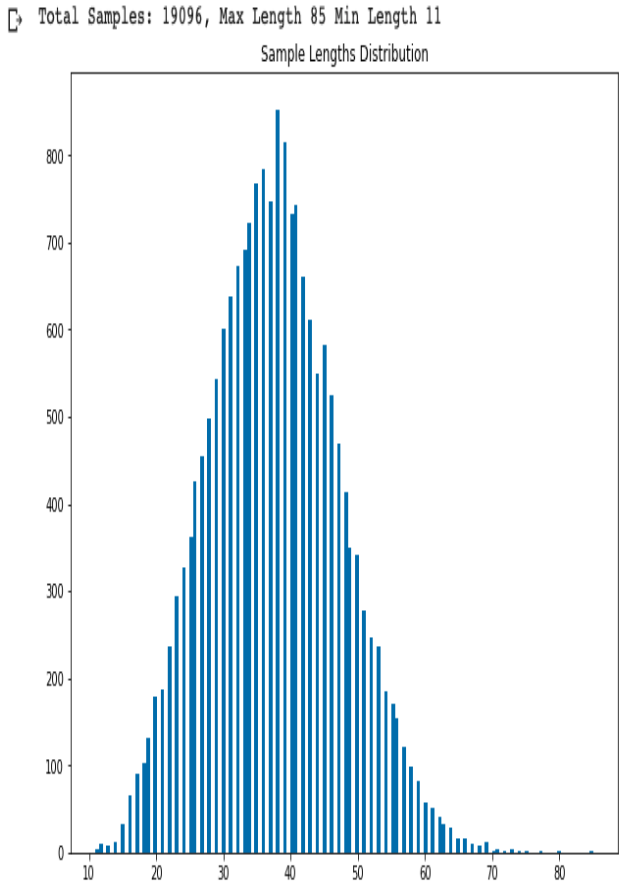

Figure 6: Length of the summary

In above figure[6] the length of the summaries has been plotted

3.3 Pre-Processing

In any machine learning task, it is important to clean and pre-process the data before feeding it to model. Textual data comes in one of the most unstructured forms, and when it comes from human its way to complicated to machine to understand. For this research, only 5k articles and summary was taken for our T5 model input and whole 19k were data point where taken for our LSTM based Encoder and Decoder model. The datasets were divided into 80-20% split for training and validation. Bengali text is very difficult to process, we need to remove the space from the word or sentence, remove unwanted special character. We must first include Bengali contractions in the dataset since contractions employ the abbreviated form of the term, whereas embedding requires the complete form of the word. The 'article' is being considered as complete text from which need to get summarize and 'Summary' is considered as our summary. Below are the steps taken for pre-processing-:
A) Removing any English character, any digits, symbols, punctuations and unwanted character in our textual data.
B) Removing the Bengali punctuation.
C) Stopwords were being removed

In this research project, LSTM based encoder and decoder and MT5 model have been

used and for that we have done two different set of Further pre-processing. Below are the steps for LSTM based encoder decoder further pre-processing steps

1) Then we have done tokenization and build Bengali vocabulary dictionary.
2) WE implemented the Genism Word2vec skipgram model with the configuration of size=300, min_count=1 and window size=10 and trained on the dataset, which have generated Bengali word vectors.
3) Finally, we will sort our summaries and article to an extent as max(len(summary))= 20, max(len(text)) = 60, max(unk count of summary) =10, max(unk count of text)= 20.

For MT5 the further preprocessing is different from the LSTM model. After cleaning the dataset we have added a new string "summarize" to the article, so that model can know the task which it has to do. As the dataset was little big to pass through the neural network, we have used DataLoader to load the data into neural network. DataLoader is much needed because we cannot feed all the data into the memory at once. We also tokenized the word with the help of pretrained MT5 tokenizer. Tokenization is been done using length parameter.

3.4 Creating Model

In this step, the selected seq2seq learning models are LSTM based encoder and decoder model and multilingual-T5 base model which we are going to fine tune it to produce Abstractive Bengali summary. All the models that we will use, will be fine tuned, and we will see the accuracy and according to the performance of the models we will be compare the models to each other to select the best model. All the models will be run for different epoch to get best result out of it.

3.5 Evaluation

Loss, Rouge 1, rouge 2, rouge L score are the evaluation metric we will be using to check the performance of the model. As well as we will compare the summary of the model that has been generated with respect to the actual summary to check whether the model is giving gold standard summary or not. Each of the models was evaluated using the summaries created by the system, and the average scores were reported. ROUGE-1: It calculates the 1-gram (per word) overlap between the system-generated and reference summaries [28]. ROUGE-2: It checks if the system generated, and reference summaries coincide in terms of bi-grams. Recall, Precision, and F-Measure are the three basic scoring methods used by ROUGE. The following formulas can be used to compute them:

1) Recall= Number of overlapping words/ Number of words in gold summary

2) Precision = Number of overlapping words/ Number of words in the reference summary

3) F-Measure = 2 X [(Precision X Recall)/(Precision + Recall)]

4 Design Specification

In this research work a Four tier framework has been implemented for the summarization of the Bengali text using NLP with LSTM-RNN based models and Transformer based model as been displayed in below figure. As shown in figure[7]various stages have been Implemented in this research project and the four tiers consist of a data layer, then data cleaning layer in which we have done our EDA and data pre-processing and then modelling layer in which we have trained our model and produce summarized text and in the last step we evaluated the model on the basis of Rouge metric.

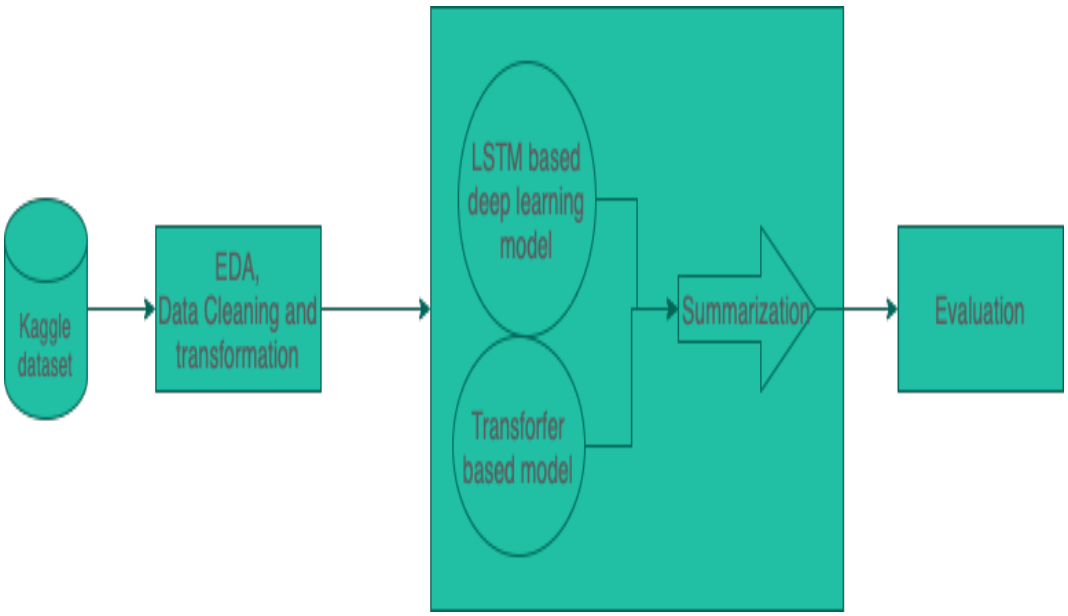

Figure 7: Frame work that has been Implemented

4.0.1 LSTM based encoder and decoder model

Our aim for the word level training for multi-language models is to optimize the predictions of the next token[Ranzato et al. (2015). In the case for abstractive summarization, given a source article x as input, a seq2seq model outputs a summary y with the probability P(y—x), where denotes model parameters (e.g., weights W and bias b). P(yt — y¡t ,x) is the conditional probability of the next token yt given all previous tokens represented by y¡t = (y1, y2,..., yt1) and source x in a neural language model (Bengioetal.2003). The text generating process may be expressed intuitively as follows:

• The decoding procedure begins with a special token 'GO¿' (start of sequence), after which the model generates a token yt with the probability P (yt — y¡t , x) = Pvocab,t(yt) at a time t.
• By greedy search, this token can be obtained that is yt = argmaxyt Pvocab,t or we can simple use sampling method.
• The produced token is used in the following stage of decoding. Unless the model emits a 'EOS' (end of sequence) token or the user-defined maximum threshold length is achieved, the generation continues.

Algorithm 1: Beam search algorithm for decoding the basic attention-based seq2seq models.

Input: Source article x, beam size B, summary length T, model parameters θ;
Output: B-best summaries;

Initialize:
Output sequences $Q^{\text{seq}} = [\text{SOS}]_{B\times T}$;
Accumulated probabilities $Q^{\text{prob}} = [1.0]_{B\times 1}$;
The last decoded tokens $Q^{\text{word}} = [\text{SOS}]_{B\times 1}$;
States (hidden and cell states for LSTM) $Q^{\text{states}} = [0.0]_{B\times|h_t^d|}$;
Context vectors $Q^{\text{ctx}} = [0.0]_{B\times|z_t^e|}$;

Compute $(h_1^e, h_2^e, ..., h_J^e)$ with encoder;
Update Q^{states} with encoder states;
for t=1, T **do**
  Initialize candidates $Q^{\text{cand,seq}}$, $Q^{\text{cand,prob}}$, $Q^{\text{cand,word}}$, $Q^{\text{cand,states}}$, $Q^{\text{cand,ctx}}$ by repeating Q^{seq}, Q^{prob}, Q^{word}, Q^{states} and Q^{ctx} B times, respectively;
  for b=1, B **do**
    Compute $P_\theta(y_{t,b}^{\text{cand}}|y_{<t,b}, x)$ using decoder LSTM cell with input $(h_1^e, h_2^e, ..., h_J^e)$, Q_b^{word}, Q_b^{states} and Q_b^{ctx};
    Select the top-B candidate words $y_{t,b,b'}^{\text{cand}}$, where $b' = 1, 2, ..., B$;
    Select corresponding probability $P_\theta(y_{t,b,b'}^{\text{cand}}|y_{<t,b}, x)$, hidden states $h_{t,b,b'}^d$, cell states $c_{t,b,b'}^d$ and context vector $z_{t,b,b'}^e$;
    Update elements of $Q_{b',b,t}^{\text{cand,seq}}$, $Q_{b',b}^{\text{cand,word}}$ with $y_{t,b,b'}^{\text{cand}}$;
    Update elements of $Q_{b',b}^{\text{cand,states}}$ with $h_{t,b,b'}^d$ and $c_{t,b,b'}^d$;
    Update elements of $Q_{b',b}^{\text{cand,ctx}}$ with $z_{t,b,b'}^e$;
    Update $Q_{b',b}^{\text{cand,prob}}$ with Eq.(77);
  end
  Flatten $Q^{\text{cand,prob}}$ and choose B best hypotheses;
  Update Q_t^{seq}, Q^{prob}, Q^{word}, Q^{states}, Q^{ctx} with corresponding candidates.
end

Figure 8: Pseudo Code for LSTM

To learn the model parameters we have used end to end cross entropy i.e. 0 and 1.For better understanding refer to the Sudo code of the Algorithm in figure[8]

Functioning -:

Firstly, the entire dataset was imported. Then the dataset was treated by removing stopwords and any other symbol and character are been removed. On the Bengali news dataset that we utilized in our trials, we employed 300-dimension word embeddings pre-trained by the word2vec method (Mikolov et al. (2013)), and we allowed the embeddings to be changed during training. After getting the word embedding, we have split the dataset for training and validation purpose. After that, the train dataset was passed through the model. The encoder is made up of two bidirectional LSTM-RNNs, each with a 400-state hidden state dimension. We have tried the to change the number of BiLSTM encoder layer from 2-3 and for decoder layer we consist an unidirectional LSTM-RNN with the same hidden state size and an attention mechanism over the source hidden state and we have put softmax layer over the target vocabulary so that it can generate words. The decoder's output is restricted to the summary's maximum length, which is set by the user. Because the target vocabulary is substantially smaller, we maintained the source and target vocabularies distinct for computational efficiency. For optimizer we have used Adam optimizer for our training model with learning rate of 0.001. For every epoch, we randomly shuffled the training data with a batch size of 32, 64. Dropout probability was also utilized, with values ranging from 0.5 to 0.7. Gradient clipping was also used

to decrease the gradient explosion of RNN networks. We also used early stopping based on the validation set and reported all performance metrics using the best model on the validation set. To construct summaries, we employed a beam size of 10 for the beam search decoder. After that we checked our model performance on testing dataset. After that machine summaries were compared with the actual summaries and we evaluated those summaries with the Rouge metric to see the accuracy at which model is building the summary.

4.0.2 MT5 Transformer Base Model

A MT5 base is on T5 model and work like T5 recipe which improve upon T5 by using GeGLU nonlinearities. As in condition with T5 model, we will be using SentencePiece model to get trained in our language sampling. The MT5 I already been pretrained on C4 corpus which has Bengali in their training set. MT5 is an encoder- decoder model and has roughly twice as parameter than just encoder only models like XLM-R, MT5 has been trained on over 1billion parameter whereas XLM-R only trained for 550 million and still the computation cost of both model is same. In mT5, the decoder usually generates two extra tokens: a class label and a sequence end token. The computational cost of classification using mT5 is generally T + 2 tokens, compared to T + 1 for an encoder-only model, because the decoder has the same architecture as the encoder (ignoring encoder-decoder attention). Encoder-decoder architectures, on the other hand, offer the advantage of being suitable to generative tasks such as abstractive summarization or conversation. At figure[9] the T5 artchiture has been briefly explained

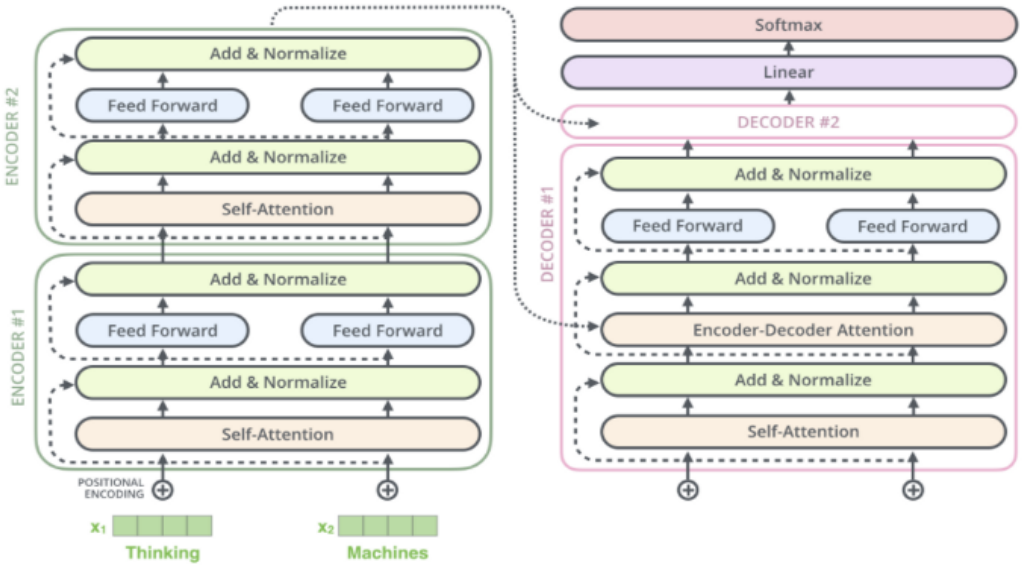

Figure 9: T5 model structure

Functioning -:

To implement the MT5 model, Firstly, the entire dataset was imported. Then the dataset was treated by removing stopwords and any other symbol and character are being removed. After that we have made a custom dataset class, this dataclass take the dataset

and tokenize the data. To tokenize the data we have chosen the pretrained MT5 tokenizer, which uses batch encode plus method to perform tokenization which will produce the necessary outputs which are Source id and source mak for the article and target id and target mask for the summary. We also have used dataloader, which helps us to create training and validation dataloader so that data can be enter into the the neural network in a systematic matter. After that we have split the dataset into 80:20, 80% is for training and 20% for validation. We will use 80% training data to fine tune our model. First we have defined the EPOCH, an epoch defines that how many times the whole dataset will be passed through the network, we have decided to train our model for 4 epoch. The batch-size and max_len parameters are used to accomplish this control the data which passes the model. We have set our input batch-size to 2 for the training and 4 for the validation, and also we have kept our MAX_LEN to 64. The language model labels are calculated from the target ids and also attention mask and source ids are been extracted. When the model produce outputs for the first element it gives out the loss for the forward pass. That loss value then is used to optimize the weights of the neuron in the network. For every 200 steps the loss value are been printed in console. We have also set are learning rate to 0.0001 with ADAM optimizer. To construct summaries, we employed a beam size of 2 for the beam search decoder. After that we checked our model performance on testing dataset. After that machine summaries were compared with the actual summaries and we evaluated those summaries with the Rouge metric to see the accuracy at which model is building the summary.

5 Implementation

This section is more focussed on discussing the implementation of the MT5 model and the LSTM-RNN based encoder decoder model for summarizing the Bengali News document. We have fine-tune both of the model to get the best result out of the model. The pre-processing of both the model are little different as in LSTM Based model we have used Word2vec for word embedding and in case for MT5 model we have used MT5 tokenizer to tokenize the words before we input it to model. All the model have been using Adam as their optimizer. MT5 have also failed to run the model due to memory allocation, for this reason we have implemented data-loader, so that the data which we are feeding to the model can be controlled. We also fine-tuned the data-loader so that our GPU space doesn't max out, we have done this by changing the Number of workers(Number of worker decide how many data should be trained simultaneously). Batch size and maximum text length were also changed, so that our GPU memory doesn't get exhausted. In this step we will see whether state of the art MT5 base model outperform the LSTM based encoder and decoder model.

5.1 Implementation of LSTM based Encoder and decoder Model

The architecture of the LSTM based Encoder and decoder consist of an encoder which is made up of two Bidirectional LSTM-RNN with each having 400 hidden state dimensions. The number of encoder layer is 2 and for decoder layer there is unidirectional LSTM-RNN with same 400 hidden state with an attention mechanism over the hidden state and also a softmax layer over the target vocabulary to generate words. Dropout probability was also utilized, with values ranging from 0.5. We ran the training model for 10 epoch with batch size of 32. In decode time, the maximum length of the summary in the

dataset is 62 words, we utilized a beam search of size 20 to construct the summary at decode time, and we limited the size of the summary to a maximum of 30 words. Using the evaluation script, we present Precision, Recall, and F1-scores from the full-length versions of Rouge-1, Rouge-2, and Rouge-L.

5.1.1 Hypertuning the LSTM based Encoder and decoder Model

In our experiments, we altered hyperparameters such as the number of epochs to , the number of RNN layers, the batch size, and the dropout probability. We have noticed that the model which have 3 layers performed well and given good Rouge scores than 2 layer model. We also noticed that increasing the number of epochs and setting a dropout probability of 0.5 and a batch size of 64 led in higher overall scores than the other tuning parameters.

5.2 Implementation of transformers based MT5 base model

In this section we will be discussing the training details for our MT5 model. First we have defined the Epoch for 5 with batch size of 4. We also used Data-loader so that we can load our dataset into neural network in a defined way. Because all of the data from the dataset cannot be put into memory at the same time, the quantity of data stored into memory and subsequently delivered to the neural network must be managed. We set number of workers to 2. The language model labels are calculated from the target ids and also attention mask and source ids are been extracted. When the model produce outputs for the first element it gives out the loss for the forward pass. That loss value then is used to optimize the weights of the neuron in the network. For every 200 steps the loss value are been printed in console.

5.2.1 Hyper parameter tuning of Base MT5 model

In our experiment, we have altered hyperparameters such as Number of epoch to 2, batch size to 2 and number of worker in data-loader to 1. We have encountered memory allocation error multiple times due to various factor, mainly because of the batch size and number of workers. For this we have hyper tuned the parameter as well as reduced the size of the data for the training and testing. We have noticed that model which has batch size of 2 and with epoch 2 have given overall higher score than other parameter tuning.

6 Evaluation

In this section we will be comparing the our model with each other on the basis of Rouge metric and will also compare the summary produced by the model. Using the evaluation script, we present Precision, Recall, and F1-scores from the full-length versions of Rouge-1, Rouge-2, and Rouge-L. We give the unseen data (Testing Dataset), trained model, tokenizer, and device characteristics to the function to execute the validation run during the validation step. This phase creates a fresh summary for any datasets that were not used during the training session.

6.1 LSTM based encoder and decoder Model with attention mechanism

In decode time, the maximum length of the summary in the dataset is 62 words, we utilized a beam search of size 10 to construct the summary at decode time, and we limited the size of the summary to a maximum of 30 words. When the hyper-tuning was done to the model, we noticed that with increase of the number of layer have increased model performance and also if when the epoch were increased its been seen that the model have outperformed the other model with less epoch cycle. It was also observed that model with 3 layer achieving the F1-score of 75% and have outperformed the 2 layer bi-directional LSTM model where it only got 63.68%. We also have plotted the loss for our best model which will give us better understanding on figure[9]

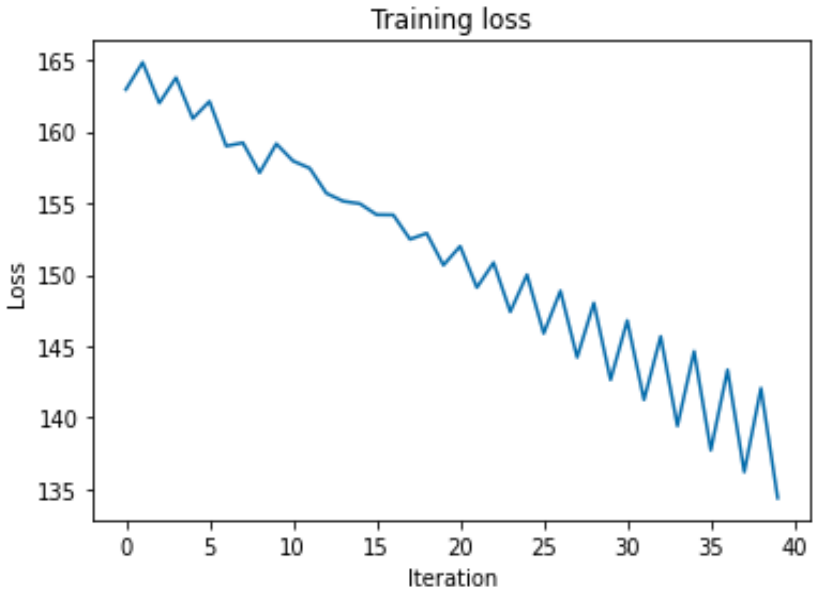

Figure 10: Loss plot for LSTM Model

6.2 MT5 Base model

In decode time, the maximum length of the summary in the dataset is 62 words, we utilized a beam search of size 5 to construct the summary at decode time, and we limited the size of the summary to a maximum of 20 words. It is been observed that model hyper-tuned Mt5 model with batch size of 2 and epoch is 2 with F1 score of 56.27% have outperformed the MT5 base model with batch size of 4 and with epoch 5 with F1 score of 48.67%.

6.3 Discussion

Similar Abstractive summarization has been presented by Ahuir et al. (2021) and their study has achieved highest F1 score 95% using MT5 base model. This Research paper tried to improve the summarization model using transformer based models but failed to do so. There might be the case that due to lack of data and computational power we failed to outperform the previous studies. But we have created base model for further

research. We will discuss more about how to we can perform well in our Future work Section

Table 2: A table caption.

Model	**Description**	**F1-score**
Previous Study	MT5 model	95%
Model 1	LSTM model with attention based	63.68%
Model 2	Tuned LSTM model with attention based	75%
Model 3	Fine-tuned MT5 model	56.27%
Model 4	MT5 base model	48.67

7 Conclusion and Future Work

In this Research paper we used a sequence-to-sequence encoder-decoder deep learning architecture as well as transformer-based model to address the challenge of abstractive text summarization for the low-resource South Asian language Bengali. We also used the popular ROUGE measure to assess the summary generation. The Study observed that the performance of the Transformer based model is little less than the Bidirectional LSTM model. In this study, both of the models were fine-tuned, and the best model were chosen as our final model. As Bengali is very low resource language and many research is going on the field of Bengali NLP application. Due to low resource language many of the phases of this research study lacked on finding the resources. The biggest problem which might have caused our models lack performance is the data source, the data source is too little, particularly on the length of summary and article were little short and thus model might have trouble to understand the pattern in the dataset. Second reason for our model to perform so poorly is computational power, as the model was running on Google colab and whenever we input large batch size the GPU memory got Max out and thus model get crashed. We tried to retify that thing with the Data loader, but then too the memory crashed keep on happening. In future, a custom dataset can be created with large article and their summary, and a custom transformer-based architecture can be built that specifically for the Bengali summarization. Also, as we only explored supervised learning field, there can be a reinforcement learning model which could apply in the dataset.

References

Ahuir, V., Hurtado, L.-F., González, J. Á. and Segarra, E. (2021). Nasca and nases: Two monolingual pre-trained models for abstractive summarization in catalan and spanish, *Applied Sciences* **11**(21): 9872.

Canete, J., Chaperon, G., Fuentes, R., Ho, J.-H., Kang, H. and Pérez, J. (2020). Spanish pre-trained bert model and evaluation data, *Pml4dc at iclr* **2020**: 2020.

Chandro, P., Arif, M. F. H., Rahman, M. M., Siddik, M. S., Rahman, M. S. and Rahman, M. A. (2018). Automated bengali document summarization by collaborating individual word & sentence scoring, *2018 21st International Conference of Computer and Information Technology (ICCIT)*, IEEE, pp. 1–6.

Cohn, T., He, Y. and Liu, Y. (2020). Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Das, A. and Bandyopadhyay, S. (2010). Topic-based bengali opinion summarization, *Coling 2010: Posters*, pp. 232–240.

de Vries, W., van Cranenburgh, A., Bisazza, A., Caselli, T., van Noord, G. and Nissim, M. (2019). Bertje: A dutch bert model, *arXiv preprint arXiv:1912.09582* .

Efat, M. I. A., Ibrahim, M. and Kayesh, H. (2013). Automated bangla text summarization by sentence scoring and ranking, *2013 International Conference on Informatics, Electronics and Vision (ICIEV)*, IEEE, pp. 1–5.

Gonzalez, J. A., Hurtado, L.-F. and Pla, F. (2021). Twilbert: Pre-trained deep bidirectional transformers for spanish twitter, *Neurocomputing* **426**: 58–69.

Haque, M. M., Pervin, S. and Begum, Z. (2016). Enhancement of keyphrase-based approach of automatic bangla text summarization, *2016 IEEE Region 10 Conference (TENCON)*, IEEE, pp. 42–46.

Inui, K., Jiang, J., Ng, V. and Wan, X. (2019). Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Joshi, A., Fidalgo, E., Alegre, E. and Fernández-Robles, L. (2019). Summcoder: An unsupervised framework for extractive text summarization based on deep auto-encoders, *Expert Systems with Applications* **129**: 200–215.

Le, H., Vial, L., Frej, J., Segonne, V., Coavoux, M., Lecouteux, B., Allauzen, A., Crabbé, B., Besacier, L. and Schwab, D. (2019). Flaubert: Unsupervised language model pre-training for french, *arXiv preprint arXiv:1912.05372* .

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V. and Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, *arXiv preprint arXiv:1910.13461* .

Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M. and Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation, *Transactions of the Association for Computational Linguistics* **8**: 726–742.

Martin, L., Muller, B., Suárez, P. J. O., Dupont, Y., Romary, L., de La Clergerie, É. V., Seddah, D. and Sagot, B. (2019). Camembert: a tasty french language model, *arXiv preprint arXiv:1911.03894* .

Mihalcea, R. (2004). Graph-based ranking algorithms for sentence extraction, applied to text summarization, *Proceedings of the ACL interactive poster and demonstration sessions*, pp. 170–173.

Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into text, *Proceedings of the 2004 conference on empirical methods in natural language processing*, pp. 404–411.

Nallapati, R., Zhai, F. and Zhou, B. (2017). Summarunner: A recurrent neural network based sequence model for extractive summarization of documents, *Thirty-First AAAI Conference on Artificial Intelligence*.

Nallapati, R., Zhou, B., Gulcehre, C., Xiang, B. et al. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond, *arXiv preprint arXiv:1602.06023* .

Pires, T., Schlinger, E. and Garrette, D. (2019). How multilingual is multilingual bert?, *arXiv preprint arXiv:1906.01502* .

Qi, W., Yan, Y., Gong, Y., Liu, D., Duan, N., Chen, J., Zhang, R. and Zhou, M. (2020). Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training, *arXiv preprint arXiv:2001.04063* .

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W. and Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer, *arXiv preprint arXiv:1910.10683* .

Rush, A. M., Chopra, S. and Weston, J. (2015). A neural attention model for abstractive sentence summarization, *arXiv preprint arXiv:1509.00685* .

Sarkar, K. (2012). An approach to summarizing bengali news documents, *proceedings of the International Conference on Advances in Computing, Communications and Informatics*, pp. 857–862.

Sarkar, K. (2014). A Keyphrase-Based Approach to Text Summarization for English and Bengali Documents, *International Journal of Technology Diffusion (IJTD)* **5**(2): 28–38. **URL:** *https://ideas.repec.org/a/igg/jtd000/v5y2014i2p28-38.html*

See, A., Liu, P. J. and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks, *arXiv preprint arXiv:1704.04368* .

Srivastava, N. and Gupta, B. K. (2014). An algorithm for summarization of paragraph up to one third with the help of cue word comparison, *International Journal of Advanced Computer Science and Application (IJACSA)* **5**: 167–171.

Uddin, M. N. and Khan, S. A. (2007). A study on text summarization techniques and implement few of them for bangla language, *2007 10th international conference on computer and information technology*, IEEE, pp. 1–4.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. and Polosukhin, I. (2017). Attention is all you need, *Advances in neural information processing systems*, pp. 5998–6008.

Virtanen, A., Kanerva, J., Ilo, R., Luoma, J., Luotolahti, J., Salakoski, T., Ginter, F. and Pyysalo, S. (2019). Multilingual is not enough: Bert for finnish, *arXiv preprint arXiv:1912.07076* .

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A. and Raffel, C. (2020). mt5: A massively multilingual pre-trained text-to-text transformer, *arXiv preprint arXiv:2010.11934* .

Zhang, J., Zhao, Y., Saleh, M. and Liu, P. (2020). Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, *International Conference on Machine Learning*, PMLR, pp. 11328–11339.

Zhong, M., Liu, P., Chen, Y., Wang, D., Qiu, X. and Huang, X. (2020). Extractive summarization as text matching, *arXiv preprint arXiv:2004.08795* .