# Abstractive and Unsupervised Bengali Text Summarization

**Sumaia Aktar**
Brock University
St. Catherines, Ontario
sa21xt@brocku.ca

## Abstract

Our motivation for this work is based on the observation that while there has been significant research on abstractive text summarization, there are only a few models designed specifically for Bengali. However, the existing models either generate summaries with limited words or no new words, meaning containing only the words from the source texts. This can make it challenging to fully comprehend the original text. Additionally, the performance of abstractive summarization systems is dependent on a large collection of document-summary pairs, which is not available for low-resource languages like Bengali. Furthermore, the sequence-to-sequence models used in these systems may struggle with long input sentences, resulting in the loss of important information. Therefore, we propose a graph-based unsupervised abstractive summarization system for Bengali text documents that can generate summaries using a Part-Of-Speech (POS) tagger, a pre-trained language model trained on Bengali texts, and one fine-tuned model. We use two readily available open-source datasets from this work (Chowdhury et al., 2021) to evaluate our model. Additionally, we fine-tuned one pre-trained language model preprocessing the datasets stated above. These two models can serve the users with the purposes of text summarization although our main focus is the unsupervised one and future plan is to work more on the supervised one.

## 1 Introduction

The process of creating a shortened version of a longer text, preserving only the most important information is known as Automatic Text Summarization. A good summary can be defined as coherent, grammatically readable, and non-redundant. Summarizations are of two types: abstractive and extractive. Extractive summarizations rank the most relevant sentences from the original text whereas abstractive summarization creates sentences that resemble human-like writing utilizing techniques of natural language generation. Neural sequence-to-sequence (seq2seq) models with attention have been successful in providing effective text generation which has been applied extensively for abstractive summarization of English language documents ((Nallapati et al., 2016); (Chopra et al., 2016); (Miao and Blunsom, 2016);(Paulus et al., 2017); (Nayeem et al., 2019)). These models are trained usually with a great deal of gold summaries, but there is a lack of extensive human-annotated abstractive summaries for low-resource languages like Bengali. On the other hand, the unsupervised approach eliminates human costs and effort for gathering and annotating large quantities of training data. That is the reason, there's a necessity for an unsupervised approach. There's the very first work in Bengali text summarization (Chowdhury et al., 2021) using unsupervised and abstractive techniques but the generated summaries contain no new words. Hence, we choose to create an effective Bengali summarizer using an unsupervised approach that would overcome these issues. Additionally, we fine-tuned one pre-trained language model with the datasets from this work ((Chowdhury et al., 2021)) allowing unsupervised Bangla text summary to compare with this supervised one and making the users decide to accept the best one. The summary of our contributions:

- According to our knowledge, our Bengali Text Summarization model (AUBTS) is the very first one that generates a summary with new words and sentences from a Bengali text single document using an unsupervised approach while being simple yet robust.
- We also provide another supervised model to allow users to generate/compare their summary in both supervised and unsupervised ways and accept the best one.
- Our unsupervised model only requires a POS

tagger, a pre-trained, and one fine-tuned language model, which is reproducible.

- Our supervised model requires a pre-trained model and two readily available open-source datasets ((Chowdhury et al., 2021))

## 2 Related works

There are a lot of works on text summarization where researchers introduced various extractive and abstractive methods. Nevertheless, there's been only a few attempts for Bengali text summarization although Bangla is the 7th most spoken language. (Bhattacharjee et al., 2020) proposed a seq2seq-based LSTM network model on Bengali abstractive news summarization with attention to the encoder-decoder. However, this model resulted in the repetition of summaries and inaccurate reproduction of factual details for long input sequences.

(Hossain, 2019) proposed a method that combined a set of mathematical rules and Bengali grammatical rules. The main contributions of their paper were sentence relevancy, sentence meaning analysis, sentence joining, and eliminating odd sentences. The steps included in the methodology like finding sentence relevancy and sentence position appreciating which might meet the challenge of Redundancy and relevancy of the summary but it is unclear how the sentence joining and sentence positioning steps contribute to the overall summary.

(Talukder et al., 2019) proposed a sequence-to-sequence RNN approach that can generate meaningful and understandable summaries from Bengali text documents. During the experiment, they were successful to accomplish their goal to reduce the training loss to .008 but generate a fluent short Bangla summary containing limited words. (Chowdhury et al., 2021) proposed the very first unsupervised model to generate abstractive summary from Bengali text documents but the limitation is that it cannot generate new words, meaning the sentences included in the generated summary are the words from the input documents only. It also does not employ grammatical correctness.

(Clarke and Lapata, 2008);(Filippova, 2010) took the very first intermediate step towards abstractive summarization which uses original sentence compression techniques for summary generation. (Filippova, 2010) proposed the first word-graph-based approach that requires only a POS tagger and a list of stop-words. Later, this approach is improved by (Boudin and Morin, 2013) by re-ranking the compression paths as per key phrases which generated informative sentences. (Nayeem et al., 2019)introduced an unsupervised summarization system that performs sentence fusion and paraphrasing jointly.

## 3 Methodology

In this section, we describe each of the steps involved in our AUBTS (Abstractive Unsupervised Bengali Text Summarization) and fine-tuned model. For acquiring a more exact representation of the information, we use BNLP[1] and NLTK[2] to preprocess sentences one by one. We perform tokenization, stop-word removal, filter punctuation marks, and Part-Of-Speech (POS) tagging.

### 3.1 Sentence clustering

This step allows us to group similar sentences for a given document. To ensure comprehensive coverage of the document and to avoid redundancy this step selects at most one sentence from each cluster (Nayeem and Chali, 2017). we measure the cosine similarity between the sentence vectors achieved from ULMfit pre-trained language model (Howard and Ruder, 2018). We calculate hierarchical agglomerative clustering using the ward's method (Murtagh and Legendre, 2014). There will be a minimum of 2 and a maximum of n-1 clusters. Here n is the number of sentences in each document. We use silhouette value to measure the number of clusters for the source document.

$$\text{Silhouette Score} = (x - y)/\max(x, y)$$

Where x and y denote the mean distance to the next closest and intra-cluster respectively.

### 3.2 Word Graph (WG) generation

The use of textual graphs in the generation of abstractive summaries provides effective results ref-(Ganesan et al., 2010). We chose the sentence Fusion technique to construct an abstractive summarizer for the Bengali Language by generating word graphs ref-(Boudin and Morin, 2013; Filippova, 2010). This method requires needs only a POS tagger and is wholly unsupervised which is highly appropriate for the low-resource setting. We construct a word graph following ref-(Boudin and Morin, 2013; Filippova, 2010), given a cluster

[1] https://bnlp.readthedocs.io/en/latest/
[2] https://www.nltk.org

of related sentences. For example, we have a set of sentences S = S1, S2,....Sn. To generate a graph G = (V; E), we iteratively add sentences to it. The words along with the parts-of-speech (POS) tags are represented as vertices and directed ages are constructed by linking up the adjected words from the sentences. After the first sentence is included in the graph, words with a similar POS tag from the other related sentences are mapped onto the node in the graph. Each of the sentences of every cluster is linked with a dummy start and end node marking the beginning and end of sentences. After generating the word graph, we can construct N-shortest paths from the start node to the dummy end node like in Figure 1.

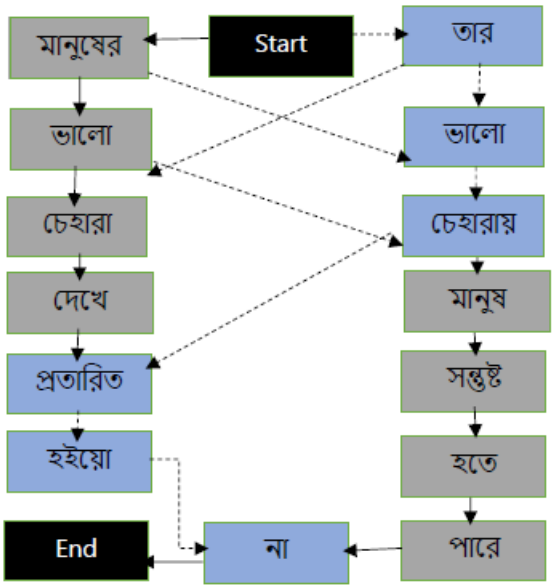


Figure 1: Example of WG for two related sentences.

Figure 2 presents two sentences, which are one of the source document clusters, and the possible paths with their weighted values are generated using the word-graph approach. Figure 1 illustrates an example WG for these two sentences. After constructing the word graph, for the two related sentences as shown in Figure 1, we get abstractive fusions (five sentences) as shown in Figure 2. Figure 2 also represents the possible weighted paths using the ranking strategy (Boudin and Morin, 2013), for these two related sentences for the same cluster. For the summary generation, we take the top-scored sentence from each cluster. The summary is then formed by merging all the top-ranked sentences. A detailed illustration of our model framework along with an example document is also presented in the Appendix section.

| Generated Sentences with the scores | |
|---|---|
| Sentences | Score |
| মানুষের ভালো চেহারায় মানুষ সন্তুষ্ট হতে পারে না [people can't be satisfied at their own good faces] | 0.912 |
| মানুষের ভালো চেহারায় প্রতারিত হইয়ো না [Don't be fooled/cheated at people's good appearances] | 0.943 |
| **তার ভালো চেহারায় প্রতারিত হইয়ো না** [don't get cheated at his/her good faces] | **1.329** |
| মানুষের ভালো চেহারা দেখে প্রতারিত হইয়ো না [Don't get fooled/cheated seeing people's good appearances] | 0.789 |
| তার ভালো চেহারায় মানুষ সন্তুষ্ট হতে পারে না [he/she can' be satisfied at his/her good appearances] | 1.112 |
| Sentences from Cluster n | |
| মানুষের ভালো চেহারা দেখে প্রতারিত হইয়ো না। [he/she can' be satisfied at his/her good appearances] | |
| তার ভালো চেহারায় মানুষ সন্তুষ্ট হতে পারে না। [People cannot be satisfied with his/her good looks] | |

Figure 2: Output of WG, given two sentences of the same cluster. Bold sentence is the top-ranked sentence to be included in the summary

### 3.3 paraphrase generation

Paraphrasing is the act of rephrasing or restating text or speech in one's own words while maintaining the original meaning. This step is crucial because it helps to avoid repetition and improve the overall coherence and readability of the summary. Using this technique, a summary can be more engaging and easier to understand for readers. We use the publicly available (Akil et al., 2022) model to generate paraphrases for each of the sentences from the summary generated in section 3.2. Before feeding into the model, we need to go through some preprocessing steps that are detailed in the 'problem set up' section. The output of the model is the final summary of our AUBTS model.

### 3.4 Fine-tuning our model

We fine-tuned Google/mt5-small [3] pre-trained model for the Bangla text summarization. We decided to proceed with fine-tuning this model as it has been pre-trained on various languages and tasks, has a strong ability to understand Bengali language nuances, and has achieved impressive results in natural language processing tasks such as text summarization (Xue et al., 2021). Our train-

[3] https://huggingface.co/google/mt5-small

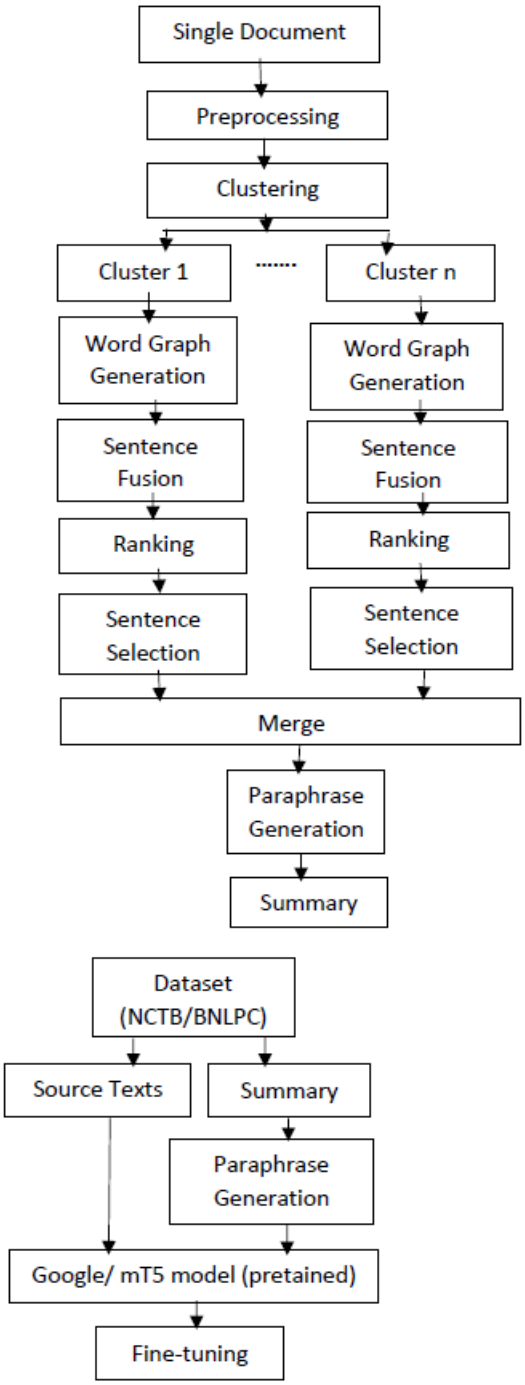


Figure 3: Overview of our models

dataset was NCTB and BNLPC (Chowdhury et al., 2021). We had to perform some preprocessing before feeding into the model which is shortly described in the experimental setup section along with detailed configuration and problem setup necessary for fine-tuning. This section outputs the abstractive summary in a supervised setting. To sum up, sections 3.3 and 3.4 are the final two models developed in our work where users would be facilitated to perform Bengali text summarization in both unsupervised and unsupervised settings and accept the best summary comparing these two. The overall process is depicted in Figure 3. For the fine-tuned model, due to some challenges faced during fine-tuning, although the evaluation score is not that much satisfactory, our unsupervised model outperforms two of the baselines for one dataset as shown in Table 2. A sample example of our AUBTS is shown in Figure 4. Also a detailed illustration of our framework is presented in the Appendix section (Figure 5)

**System Summary (AUBTS)**

বোকা মানুষেরা খারাপ আচরণের মানুষের চেহারা দেখে পটে যায় এবং এইরূপ ভুল ধারনার উপরেই জীবন পার করে দেয় অন্ধের মতোন। যার আচরণ ভালো নয়, তার চলাফেরাকে মানুষ ভালো চোখে দেখে না। তার ভালো চেহারায় প্রতারিত হইয়ো না।

[Fools are misled by the outward appearance of evil-doers and live blindly on this misconception. People do not see well the movements of a person who does not behave well. We should not be deceived by his/her good appearance]

**Human Reference**

খারাপ লোকেরা মানুষের বাহ্যিক চেহারা দেখে আকৃষ্ট হয় এবং এর ফল উপভোগ করে। মানুষ তাদের স্বভাব, স্পর্শ, প্রথাকে ঘৃণা করে। আমরা আমাদের চরিত্র গড়ে তোলার জন্য কঠোর পরিশ্রম ও সাধনা করি নতুবা শয়তানকে পরাজিত করা সম্ভব নয়। তার সুন্দর মুখটা দেখে আনন্দিত হবেন না।

[Fools are blinded by the appearance of bad-mannered people and love to live their lives blindly on these misconceptions. People do not like bad-manner people. Don't be excited by seeing their good faces.

Figure 4: Example output of our AUBTS model with English translations.

## 4 Experimentation procedure

### 4.1 Environment Set Up

- CPU @ 2.30GHZ
- 8GB RAM
- conda 23.3.1
- PyCharm – 2022.3.2
- Python 3.9.13
- torch 1.7+
- NetwokX 2.6
- FastAI (support torch below 1.8)

### 4.2 System configuration

During the experiment, my operating system was Windows: - 10 with 8GB RAM and as I had previously installed Anaconda and Python on my machine, I just tried following the rest of the package installation instructions as described in the README which is the referenced paper (Chowdhury et al., 2021) of our works. By the term referenced paper, we mean the paper that we took as a base and then apply for possible extensions as described in the model section. However, the installation procedure couldn't be continued under the conda environment due to several challenges as described in the 'challenges' section. Later, the conda interpreter in the pyCharm editor was changed only to a plain Python installation directory, and the steps regarding the libraries installations that needed to be done in the conda environment were repeated. As the PyCharm editor provides an opportunity to maintain more than one interpreter for the same projects, we just need to change the interpreter directory in the settings section although much time was taken for the referenced projects to be set up every time, we change the interpreter after the installation. One more important thing is, fastAI does not support 1.8 or later versions that I came to know very later and for this reason, right after the fastAI installation, torch was downgraded automatically to 1.7 that again creating dependency conflicts for the other packages which depended on the earlier version of torch (torch 2.0.0) just right before the downgrading. Another suggestion is to allow as much space as possible in the disk (windows) where the project would be located (min 150 GB) and the availability of RAM size of your machine should be at least 16GB or more so that you can optimize time during project start-up and execution of any other module. Also, this will facilitate during training. Another massy library, NetwokX, only supports Python 3.7 to 3.9. Lastly, to minimize the number of dependency conflicts due to torch version degradation as described above, we would suggest trying installing FastAI and NetworkX first and then following the dependent installation of these two libraries during installation in the command prompt in Windows.

### 4.3 Challenges

First of all, the referenced work (Chowdhury et al., 2021) was done in the UNIX operating system. There were a significant number of files that were using the default directory location and properties of UNIX. That is the reason, all these files contained errors as our OS was Windows and we needed to fix up all these errors. Another significant thing that was missed in the README of the referenced work is that they did not mention which version to downgrade in case of getting an error although they mentioned that we might try downgrading PyTorch, FastAI, and NetworkX if we found any error. That is the reason we had to go through several uninstallation and installation procedures related to conda for more than 1 week to resolve dependency conflicts. Finally, we came to know that the 'bnlm' library which was one of the key files of the project can only be installed using the 'pip' command instead of using the 'conda' command. So, we changed the environment from conda to plain Python, and some of the existing errors were resolved. Nevertheless, we were able to run the project partially because it was tough to manually edit all the files, especially those containing UNIX's properties and system files. Therefore, we targeted the clustering, baseline algorithm evaluation part and datasets portion of the repository. We were able to find top-ranked sentences and generate rouge scores and then compared them with our models following the works sumy[4] and python-rouge [5]. For fine-tuning and evaluation, we followed this tutorial [6]. Following this one, we had been able to train and evaluate the NCTB and BNLPC datasets. However, source document files of the datasets contained a significant number of errors and we needed to process each of them manually as there were different errors for different files that were not removed after dataset preprocessing. Still, we could not fix up all as we found it time-consuming and as a result, as seen in the log files, several epochs ended up before completing all the entries. To be noted, we took screenshots to note down epoch-wise scores as sometimes during the training, the machine got unexpected shutdown. We keep this to be examined in our future work. our fine-tuned work is publicly available here [7].

### 4.4 Datasets

We used two datasets from (Chowdhury et al., 2021) i.e. NCTB (National Curriculum and Textbook Board) and BNLPC (Haque et al., 2015).

[4] https://github.com/miso-belica/sumy
[5] https://github.com/tagucci/pythonrouge
[6] https://huggingface.co/learn/nlp-course/chapter7/5
[7] https://github.com/SumaiaBristy/AUBTS

NCTB consists of 139 samples of abstractive document-summary pairs written by professional summary writers of the NCTB. The NCTB monitors the distribution and development of Bangladeshi textbooks, which are followed by the maximum number of Bangladeshi schools. From the dataset, copy rate between the human summaries and source documents is measured and it is clear from the table that these two datasets are highly abstractive and thus might be served as a robust benchmark for this kind of work in the future. Furthermore, to demonstrate our proposed work's effectiveness, an experiment is also conducted with an extractive dataset BNLPC (Haque et al., 2015). For the extractive evaluation, to compare with the baselines, the fusion part of the abstractive sentence was removed. Overall statistics of the datasets are shown in table 1.

| | NCTB | BNLPC |
|---|---|---|
| Total #samples | 139 | 200 |
| Source documents length | 91.331 | 50.75 |
| Homan reference length | 36.23 | 67.06 |
| Summary copy rate | 27% | 99% |

Table 1: Statistics of the datasets used for our experiment. Length is expressed as Avg. number of tokens.

### 4.5 Problem set up

As a first step, an account was needed to be created in hugging face for getting the access token that was a prerequisite to run two of the models as stated in the methodology section. This way we were assured to get access to all the models, datasets, and all the stuff needed. For each model, we had to define a model checkpoint which is specifically the model name, defined by hugging face. A tokenizer and a pre-trained model could easily be cloned from the hugging face just by using the model checkpoint with a structured format specified by their API that was pretty easy. We used one pre-trained model[8] for fine-tuning and one fine-tuned model (Akil et al., 2022) for paraphrase generation as our goal was not to start from scratch. However, Training arguments also needed to be fixed for the pre-trained model that is defined in the 'hyperparameter search' section. After downloading the two models from hugging face as stated above, we performed some preprocessing on the datasets before feeding to the pre-trained model. Firstly, we read the source and respective summary file for each document for a specific dataset and then perform sentence tokenization to convert each passage to a list of separate sentences. This list was then fed into the fine-tuned model to generate paraphrases in the form of tokens for Bangla text. For this process, a special normalization pipeline and tokenizer were used as suggested here [9]. The generated tokens were decoded back using this pipeline and were processed more for taking back in the form of a list of sentences. This list is then converted to a string and a data dictionary instance was created. Lastly, the final dataset was generated from the data dictionary following. For the training, this final dataset was needed to split to train, test, and validation set and we faced challenges doing so using the built-in method from sklearn library . So, we split manually. For batch size, our machine supported only 16 because setting an upper value more than that ended up with an insufficient memory error. For this reason, training took much more time to be completed. Necessary coding for all the steps mentioned above is located in this fine-tuning-file following this coding tutorial[10]

### 4.6 Hyperparameter Search

we planned fine-tuning google/mT5-small for 8 epochs tuning the learning rate for 3e-4 to 5.6e-5. But our machine didn't support that way and we could tune only for 3e-4 and 5.6e-5. We chose the best model based on the rouge score.

### 4.7 Automatic Evaluation

Our system is evaluated using an automatic evaluation metric ROUGE F1 (ref Lin, 2004) keeping no limit for words. 3-best sentences were extracted from our system and the baseline systems. To measure the summary informativeness, we report ROUGE-1 and ROUGE-2 scores (overlap of unigram and bigram) and ROUGE-L (longest common subsequence) to measure summary fluency. The way ROUGE evaluates summaries is based only on the similarity of words used in the text, so it can evaluate summaries in Bengali or any other language without requiring language-specific modification.

### 4.8 Baseline Systems

We compare our works with several well-established systems like LexRank((Erkan and

[8] https://huggingface.co/google/mt5-small

[9] https://github.com/csebuetnlp/normalizer

[10] https://huggingface.co/learn/nlp-course/chapter7/5

Radev, 2004), TextRank ((Mihalcea and Tarau, 2004)), GreedyKL ((Haghighi and Vanderwende, 2009)), and SumBasic ((Nenkova and Vanderwende, 2005) ). To be noted, these summarizers are designed for the English language and extractive. An open-source implementation was used to make these systems adapted for the Bengali language according to ((Chowdhury et al., 2021)).

### 4.9 Results

Our model's performance is reported in Table-2 in terms of F1 scores of R1, R2, and R-L. To be clarified, we trained for a total of 8 epochs and noted down the lowest rouge score as some epochs ended up with errors as described in the problem setup section. However, according to Table 2, our unsupervised model outperforms two of the baselines in terms of R-1 and R-2 and for the BNLPC although it doesn't outperform any, still it is satisfactory in the sense that the scores are not that much bad as we had to face several challenges during training as discussed in the challenges section. Also, from Table 2, it can be summarized that the AUBTS model performs best compared to our fine-tuned model. In the future, we plan for better fine-tuning and introducing a system that can jointly perform abstractive summarization in both supervised and unsupervised settings.

| NCTB | R-1 | R-2 | R-L |
|---|---|---|---|
| Random Baseline | 9.43 | 1.45 | 9.08 |
| GreedyKL | 10.01 | 1.84 | 9.46 |
| LexRank | 10.65 | 1.78 | 10.04 |
| TextRank | 10.69 | 1.62 | 9.98 |
| SumBasic | 10.57 | 1.85 | **10.09** |
| Fine-tuned mT5 (ours) | **18.22** | 1.30 | 9.3 |
| [AUBTS](ours) | **11.22** | **3.9** | 4.78 |

| BNLPC | R-1 | R-2 | R-L |
|---|---|---|---|
| Random Baseline | 35.57 | 28.56 | 35.04 |
| GreedyKL | 48.85 | 43.80 | 48.55 |
| LexRank | 45.73 | 39.37 | 45.17 |
| TextRank | **60.81** | **56.46** | **60.58** |
| SumBasic | 35.51 | 26.58 | 34.72 |
| Fine-tuned mT5 (ours) | 45.42 | 28.71 | 37.59 |
| [AUBTS](ours) | 55.33 | 22.34 | 15.89 |

Table 2: Results on NCTB dataset and BNLPC

### 4.10 Human Evaluation

To evaluate the abstractive summaries generated by our system, we assigned three evaluators to rate each summary based on three different aspects: Content, Readability, and Overall Quality (Chowdhury et al., 2021), as ROUGE (Lin, 2004) is biased towards surface level lexical similarities and is unsuitable for evaluating abstractive summaries; We rate each summary on a scale from 1 to 5, whereas 1 indicates poor performance and 5 represents good performance, and we obtained average scores of 3.11, 4.95, and 3.2 in content, readability, and overall quality, respectively.

## 5 Conclusion and future work

In this paper, we have developed an abstractive summarization model for both supervised and unsupervised settings given single Bengali text documents. As paraphrasing is included in the summary generation, our unsupervised model can generate a summary containing new words and sentences while enhancing the readability and overall coherence maintaining the original meaning. As discussed in the challenges section, we have still a lot of work to be done for better fine-tuning. Therefore, our main contribution is the AUBTS model that outperforms two of the well-established baselines as shown in the result section. In the future, we plan to jointly provide Bengali text summarization for supervised and unsupervised settings.

## Limitations

The main drawback of our project is that we use an already fine-tuned paraphrased model to generate paraphrases for our summary. This model although successfully generates new phrased sentences, sometimes introduces contradictory words and the original meaning is reversed. Also, the training dataset used in fine-tuning was not satisfactory as we faced challenges in preprocessing the dataset very well. As a result, the evaluation results of the fine-tuned model did not comply with the baseline standards. In the future, we plan to introduce better paraphrasing techniques along with large-scale authentic Bengali datasets for training purposes.

## References

Ajwad Akil, Najrin Sultana, Abhik Bhattacharjee, and Rifat Shahriyar. 2022. Banglaparaphrase: A high-

quality bangla paraphrase dataset. *arXiv preprint arXiv:2210.05109*.

Prithwiraj Bhattacharjee, Avi Mallick, Md. Saiful Islam, and Marium-E-Jannat. 2020. Bengali abstractive news summarization (BANS): A neural attention approach. In *Advances in Intelligent Systems and Computing*, pages 41–51. Springer Singapore.

Florian Boudin and Emmanuel Morin. 2013. Keyphrase extraction for n-best reranking in multi-sentence compression. pages 298–305.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *North American Chapter of the Association for Computational Linguistics*.

Radia Rayan Chowdhury, Mir Tafseer Nayeem, Tahsin Tasnim Mim, Md. Saifur Rahman Chowdhury, and Taufiqul Jannat. 2021. Unsupervised abstractive summarization of bengali text documents. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Online. Association for Computational Linguistics.

James Clarke and Mirella Lapata. 2008. Global inference for sentence compression : an integer linear programming approach. *J. Artif. Intell. Res.*, 31:399–429.

G. Erkan and D. R. Radev. 2004. LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Katja Filippova. 2010. Multi-sentence compression: Finding shortest paths in word graphs. In *International Conference on Computational Linguistics*.

Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. pages 362–370.

Md. Majharul Haque, Suraiya Pervin, and Zerina Begum. 2015. Automatic bengali news documents summarization by introducing sentence frequency and clustering. pages 156–160.

Md. Monowar Hossain. 2019. Automatic text summarization for bengali language including grammatical analysis. *International Journal of Scientific & Technology Research*, 8:288–292.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. pages 328–339.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. page 10.

Yishu Miao and Phil Blunsom. 2016. Language as a latent variable: Discrete generative models for sentence compression. *CoRR*, abs/1609.07317.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Conference on Empirical Methods in Natural Language Processing*.

Fionn Murtagh and Pierre Legendre. 2014. Ward's hierarchical agglomerative clustering method: Which algorithms implement ward's criterion? *Journal of Classification*, 31:274–295.

Ramesh Nallapati, Bing Xiang, and Bowen Zhou. 2016. Sequence-to-sequence rnns for text summarization. *CoRR*, abs/1602.06023.

Mir Tafseer Nayeem and Yllias Chali. 2017. Extract with order for coherent multi-document summarization. In *TextGraphs@ACL*.

Mir Tafseer Nayeem, Tanvir Fuad, and Yllias Chali. 2019. *Neural Diverse Abstractive Sentence Compression Generation*, pages 109–116.

Ani Nenkova and Lucy Vanderwende. 2005. The impact of frequency on summarization.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *CoRR*, abs/1705.04304.

Md Talukder, Sheikh Abujar, Abu Kaisar Mohammad Masum, Fahad Faisal, and Syed Hossain. 2019. Bengali abstractive text summarization using sequence to sequence rnns.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer.

# A Appendix

| Input Document |
| --- |
| মানুষের ভালো চেহারা দেখে প্রতারিত হইয়ো না। যার আচরণ ভালো নয় দেখতে সে যেমনই হোক, তার চলাফেরাকে মানুষ ভালো চোখে দেখে না। খারাপ আচরণের মানুষেরা অন্য সবার সাথে খারাপ ব্যাবহার করে যদিও তারা দেখতে সুন্দর হয়। তার ভালো চেহারায় মানুষ সন্তুষ্ট হতে পারে না। বোকা মানুষেরা মানুষের কাজ নয়, চেহারা দেখে পটে যায় এবং এইরূপ ভুল ধারনার উপরেই জীবন পার করে দেয় অন্ধের মতোন। মানুষ নিজে আচরণে সুন্দর না হলেও সে সেইরকম সৌন্দর্যকেই ভালোবাসে।<br><br>[Don't be fooled by good appearances of people. No matter how good their appearance is, people does not look favorably on his/her behavior. Bad mannered people treat everyone badly even though they look nice. People cannot be satisfied with his/her good looks. Foolish people get motivated by the appearances of evil people instead of their work and live their lives blindly based on such misconceptions. Although people themselves are not beautiful in, behavior they love such beauty.] |
| Sentence Clustering |
| **Cluster #1**<br>Sentence #1: খারাপ আচরণের মানুষেরা অন্য সবার সাথে খারাপ ব্যাবহার করে যদিও তারা দেখতে সুন্দর হয়।<br>Sentence #2: বোকা মানুষেরা মানুষের কাজ নয়, চেহারা দেখে পটে যায় এবং এইরূপ ভুল ধারনার উপরেই জীবন পার করে দেয় অন্ধের মতোন।<br>**Cluster #2**<br>Sentence #1: যার আচরণ ভালো নয় দেখতে সে যেমনই হোক, তার চলাফেরাকে মানুষ ভালো চোখে দেখে না।<br>Sentence #2: মানুষ নিজে আচরণে সুন্দর না হলেও সে সেইরকম সৌন্দর্যকেই ভালোবাসে।<br>**Cluster #3**<br>Sentence #1: মানুষের ভালো চেহারা দেখে প্রতারিত হইয়ো না।<br>Sentence #2: তার ভালো চেহারায় মানুষ সন্তুষ্ট হতে পারে না। |
| Sentence Fusion & Ranking |
| **Cluster #1**<br>বোকা মানুষেরা খারাপ আচরণের মানুষের চেহারা দেখে পটে যায় এবং এইরূপ ভুল ধারনার উপরেই জীবন পার করে দেয় অন্ধের মতোন<br>**Cluster #2**<br>যার আচরণ ভালো নয়, তার চলাফেরাকে মানুষ ভালো চোখে দেখে না।<br>**Cluster #3**<br>তার ভালো চেহারায় প্রতারিত হইয়ো না। |
| Initial Summary |
| বোকা মানুষেরা খারাপ আচরণের মানুষের চেহারা দেখে পটে যায় এবং এইরূপ ভুল ধারনার উপরেই জীবন পার করে দেয় অন্ধের মতোন। যার আচরণ ভালো নয়, তার চলাফেরাকে মানুষ ভালো চোখে দেখে না। তার ভালো চেহারায় প্রতারিত হইয়ো না।<br>[Foolish people are blinded by the appearance of bad behavior people and live their lives blindly based on such misconceptions. People do not look favorably on the behavior of those who do not behave well. Don't be fooled by his/her good looks.] |
| Final Summary After Paraphrasing |
| বোকারা খারাপ আচরণকারীদের বাহ্যিক চেহারা দেখে বিভ্রান্ত হয়ে যায় এবং অন্ধের মতো জীবন অতিবাহিত করে এই ভুল ধারণার ওপর। যে ব্যক্তি ভাল আচরণ করে না তার গতিবিধিকে মানুষ ভালভাবে দেখে না। ওর চেহারা দেখে ধোঁকা পেয় না।<br>[Fools are misled by the outward appearance of evil-doers and live blindly on this misconception. People do not see well the movements of a person who does not behave well. We should not be deceived by his/her good appearance] |