Bangla Key2Text: Text Generation from Keywords for a Low Resource Language

Tonmoy Talukder[1], **G M Shahariar**[2]
[1]Ahsanullah University of Science and Technology
[2]University of California - Riverside
tonmoytalukder.cs@gmail.com, gshah010@ucr.edu

Abstract

This paper introduces *Bangla Key2Text*, a large-scale dataset of 2.6 million Bangla keyword–text pairs designed for keyword-driven text generation in a low-resource language. The dataset is constructed using a BERT-based keyword extraction pipeline applied to millions of Bangla news texts, transforming raw articles into structured keyword–text pairs suitable for supervised learning. To establish baseline performance on this new benchmark, we fine-tune two sequence-to-sequence models, mT5 and BanglaT5, and evaluate them using multiple automatic metrics and human judgments. Experimental results show that task-specific fine-tuning substantially improves keyword-conditioned text generation in Bangla compared to zero-shot large language models. The dataset, trained models, and code are publicly released to support future research in Bangla natural language generation and keyword-to-text generation tasks.

Keywords: Transfer learning for under-resourced language processing, Resources for low-resource language, Low-resource methods for NLP

1. Introduction

Text generation is a significant and complex task within the realm of natural language generation (NLG) (Garbacea and Mei, 2020). The text generation task can be characterized as generating an expected output sequence based on a given input sequence, which is also known as sequence-to-sequence (seq2seq) modeling (Sutskever et al., 2014). In this approach, a seq2seq model uses an encoder-decoder architecture where the encoder converts the input text (whether in the form of sequence or keywords) into a fixed-sized vector, which is then mapped to the desired target sequence by the decoder. Natural language generation models have been significantly improved due to the advancement in deep learning (LeCun et al., 2015), enabling machines to understand and produce human-like language. The notion of seq2seq modeling has paved the way for the rapid advancement of natural language generation systems such as machine translation, summarization, question answering, and dialogue systems.

Keywords are concise representations of the content found within a document (Siddiqi and Sharan, 2015). They often consist of one or more words and help to capture the essence of the text. By incorporating keywords into the text generation process, semantic coherence can be preserved, resulting in higher output quality. *Keywords-to-text* generation approach is proven to be useful in several NLG tasks such as generating dialogue responses (Song et al., 2019a; Zhang et al., 2018), generating method names (Ge and Kuang, 2021), producing abstractive summaries (Li et al., 2020), and generating image captions (Yong et al., 2021).

Keys	**[B]** জমজমাট, মেলা, শুক্রবার, গতকাল **[E]** crowded, fair, Friday, yesterday
Text	**[B]** গতকাল শুক্রবার মেলা জমজমাট হয়েছে। **[E]** Yesterday was Friday and the fair was crowded.
Keys	**[B]** সাধারণ, অভ্যস্ত, যাপনে, যশোদাবেন, জীবন **[E]** simple, accustomed, living, Yashodaben, life
Text	**[B]** খুবই ধর্মপ্রাণ যশোদাবেন একেবারে সাধারণ জীবন যাপনে অভ্যস্ত। **[E]** Yashodaben is accustomed to living a simple life.

Table 1: Few examples of generated Bangla texts by our fine-tuned **mT5** model from the provided unordered keywords. **B** represents the keywords or texts in Bangla and **E** represents their corresponding English translation.

The use of keywords aids in the generation process, ensuring that the generated text remains relevant and meaningful.

Bangla is considered a low-resource language when it comes to language processing, though it is the seventh among the most spoken languages around the world [1]. While several studies on

[1]ethnologue.com/insights/ethnologue200

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 3805–3822
11–16 May 2026.

keywords-to-text generation have been conducted in languages such as English (Özbal et al., 2013; Shen et al., 2022; Nie et al., 2022; Mishra et al., 2020), Japanese (Uchimoto et al., 2002), and Chinese (Song et al., 2019b), based on our comprehensive investigation, we have not come across any existing studies on keywords-to-text generation in Bangla. Although large language models (LLMs) have achieved notable success in general-purpose text generation, we found that particularly in Bangla, these models in their quantized form, often underperform when given unordered keyword inputs without in-context learning or fine-tuning. Our approach addresses this gap through a lightweight keyword extractor and task-specific fine-tuned smaller language models that are more effective, efficient, and reproducible. In summary, we have made the following contributions:

- We introduce **Bangla Key2Text**, a large-scale dataset of 2.6 million Bangla keyword-text pairs - the first and largest resource of its kind for Bangla keyword-to-text generation (Section 3.1).

- We develop a BERT-based keyword extractor for identifying contextually salient Bangla keywords and fine-tune two encoder-decoder models, $\mathrm{mT5}$ and $\mathrm{BanglaT5}$, for generating coherent, keyword-faithful Bangla text (Sections 3.2 and 4). We show a few examples in Table 1.

- We conduct extensive evaluations, including on unseen and dialectal data, and benchmark our models against multiple 4-bit quantized open-source LLMs (LLaMA, Phi, Gemma, Mistral, Qwen, etc.), demonstrating that our fine-tuned models consistently outperform them across standard NLG metrics (Section 4.4).

- We analyze model behaviors such as cross-lingual transfer when mixing Bangla and English keywords, highlighting new opportunities for future research in multilingual keyword-driven text generation (Section 4.4).

While our work includes keyword extraction and text generation models, the primary contribution of this paper is the Bangla Key2Text dataset, which provides the first large-scale benchmark for keyword-to-text generation in Bangla. The proposed models serve primarily as baseline systems to demonstrate the usability of the dataset and establish reference performance for future research. We have made our dataset, code, and fine-tuned models publicly available[2].

[2] https://github.com/TonmoyTalukder/Bangla-Key2Text

2. Related Works

Several recent studies have explored text generation from keywords using different approaches in languages other than Bangla. Uchimoto et al. (2002) proposed a statistical tri-gram model trained on Japanese data to generate coherent keyword-based sentences. Similarly, Song et al. (2019b) proposed an attention-based encoder-decoder model trained on large Chinese dataset to generate coherent Chinese sentences from keywords. Özbal et al. (2013) introduced a rule-based approach that combines sentence fragments to generate creative sentences from keywords, using defined structural rules. Shen et al. (2022) developed an augmentative and alternative communication (AAC) system using sequence-to-sequence models with attention, which generates context-aware personalized sentences from keywords based on AAC user data. Nie et al. (2022) proposed neural models with attention mechanisms to better control the lexical complexity in keyword-to-text generation. Their method offers fine control over the use of rare vocabulary, leading to more readable and user-tailored sentences. Mishra et al. (2020) also presented a neural approach that generates text from keywords by following part-of-speech (POS)-based templates, with attention mechanisms ensuring alignment between structure and content. Park and Ahn (2018) applied a generative adversarial network (GAN) framework, where an LSTM-based encoder-decoder served as the generator and a Bi-LSTM with self-attention acted as the discriminator. Both components were trained adversarially to improve keyword-to-text generation.

3. Methodology

The proposed methodology of this work consists of three important steps: (a) dataset construction, (b) keywords extraction, and (c) text generation. We gathered 2.6 million Bangla texts from a large public dataset, and then extracted important keywords to curate a dataset of 2.6 million Bangla keywords-text pairs. After that we fine-tuned two pre-trained language models using 2 million keywords-text pairs, while a separate set of $500K$ and $100K$ keywords-text pairs were used for validation and test purpose. We provide a detailed description of each step below.

3.1. Dataset Development

To the best of our knowledge, there is currently no dataset specifically designed for the purpose of generating Bangla text from keywords. To address this gap, we developed a Bangla keywords-text

Keywords	Text
[B] শীতের, কম, আগে, সম্ভাবনা **[E]** winter, slim, before, chance	**[B]** শীতের আগে সেই সম্ভাবনা কম। **[E]** The chance is slim before winter.
[B] খবর, কোত্থেকে, বানোয়াট **[E]** news, where, fake	**[B]** তারা কোত্থেকে পেল এসব বানোয়াট খবর? **[E]** Where did they get these fake news?
[B] পানকৌড়ি, সাঁতার, হাওরের, জলে **[E]** Cormorants, swim, lake, water	**[B]** হাওরের শান্ত জলে আপন মনে সাঁতার কাটে পানকৌড়ি। **[E]** Cormorants swim at their own pace in the calm waters of the lake.
[B] ছাত্রীও, বক্তব্য, দেয়নি, নির্যাতিত **[E]** student, statement, not, abused	**[B]**এমনকি নির্যাতিত ছাত্রীও কোনো বক্তব্য দেয়নি। **[E]** Even the abused student did not give any statement.

Table 2: Few keywords-text pairs from the **Bangla Key2Text** dataset.

pairs dataset named **Bangla Key2Text** (Bangla Keywords to Text) encompassing a total of 2.6 million keywords-text pairs. We elaborate on the dataset development process below.

Although the underlying news articles are publicly available, constructing a keyword-to-text dataset at this scale requires a dedicated pipeline for keyword extraction, filtering, and dataset structuring. Our contribution lies in transforming raw Bangla text corpora into a structured dataset of keyword–text pairs suitable for supervised training and evaluation, including curated train, validation, and test splits as well as manually annotated evaluation data.

(a) Data Collection. We utilized the **Bangla Newspaper Dataset** (Nazi, 2020) that consists of 409.5K Bangla news articles. For our study, we randomly chose 2.6 million texts from those articles. In the 2.6 million data, the maximum number of sentences in a text is 67 and the minimum is 1. There are 2 to 2.25 sentences on average. We removed special characters and HTML tags. The Bangla newspaper corpus provides raw articles rather than structured keyword–text pairs. Therefore, we apply preprocessing and automatic keyword extraction to convert the texts into paired inputs consisting of unordered keywords and corresponding sentences suitable for supervised keyword-to-text generation.

(b) Data Split. We partitioned 2.6 million texts into three groups: training, validation, and test. The training set comprises 2 million texts, the validation set contains $500K$ texts, and the test set consists of $100K$ texts. In addition to that, we curated a separate evaluation dataset by manually extracting keywords from the first $10K$ texts from the $100K$ test set. The reason is twofold: (i) assess the effectiveness of our proposed keyword extraction method and (ii) determine the appropriate decoding technique for the text generation models. This manual extraction process involved two annotators, resulting in two sets of keywords for each text. We formatted each text by separating every word with a comma and presented the dataset to the annotators. They were instructed to retain only the essential keywords while discarding the less significant ones, utilizing their own discretion. Subsequently, another annotator was enlisted to choose the final set of keywords from the two sets provided by the initial annotators.

(c) Key2Text Dataset. Based on the experimental results found for keyword extraction (details in section 3.2), we employed $\mathrm{BanglaBERT}$ (Bhattacharjee et al., 2022) based extractor to extract keywords from all the 2.6 million texts (train, test and validation sets) and curated a dataset consisting of Bangla keywords-text pairs (*Bangla Key2Text*).

(d) Dataset Statistics. The maximum number of extracted keywords in a text is 259, and the average number of extracted keywords per text is 9.381. The maximum number of words in a text is 437, and the average number of words per text is 15.142. The total number of words (text length) is 40487220 and the extracted number of keywords is 24414856 over the 2.6 million data. The ratio of key word length and text length is 0.6030. The top five frequently occurring keywords in the dataset are as follows - কী (what): 272 times, সূত্র (source): 270 times, প্রতিনিধি (representative): 212 times, এএফপি (AFP): 196 times, and না (no): 191 times. We show some instances from the dataset in Table 2.

3.2. Keywords Extraction

We have developed a pre-trained BERT (Devlin et al., 2019) based keyword extractor consisting of two main steps: *keywords scoring* and *keywords selection*. Extracting nine keywords per text in average for 2.6 million data manually is a very difficult task which necessitates the development of automatic extraction systems. Moreover, as the extracted keywords are contained in the text, rather than random selection, we tried to extract those that contribute most to the context of the text. Our proposed keyword extractor first assigns importance score to each word in a text, then select and extract keywords with top scores. The extracted keywords are returned in an unordered format.

3.2.1. Experimental Settings

To evaluate the effectiveness of our keyword extraction method, we compare it with both neural and traditional keyword extraction baselines, including two Bangla BERT-based models and two widely used unsupervised methods (TextRank and YAKE). The BERT based models were utilized for inference using their default hyper-parameters in the keyword extraction phase. No fine-tuning was performed. The tokenized text is passed through the pre-trained model to obtain the *last hidden state* as the token embeddings, which had a shape of 512×768. By computing the average of the embeddings across the 512 tokens, we derived a mean embedding of shape 768.

3.2.2. Keywords Scoring

To assign importance score, we utilized token embeddings using the pre-trained BERT model. We first calculated the average of all word embeddings in a text. Then for each word, we measured cosine similarity score between the word embedding and mean embedding which we considered as the importance score. We observed that the embeddings of important words are closer to the mean embedding. A high cosine similarity score between a word's embedding and the mean embedding of the text indicates that the word's embedding is pointing in a similar direction as the mean embedding of the text. In other words, the word's context is similar to the overall context of the text. This suggests that the word is closely related and contributes most to the main topic or theme of the text. On the other hand, a low cosine similarity score indicates that the word's embedding and the mean embedding are pointing in different directions. This suggests that the word's context does not align well with the overall context of the text and the word is less central to the main theme of the text. As the BERT tokenizer uses sub-word tokenization, so a single word can get tokenized into different tokens. To address this, we accumulate all the sub-words to form the original word and consider the average of all the sub-word token embeddings as the word embedding. As an example, we consider the word "মোজাফফর" ("Mozaffar" which is the name of a person). During tokenization, the word gets tokenized into three sub-word tokens: মোজা, ##ফ and ##ফর. At first, we get the embedding for each of these three sub-tokens. We store the embedding of the first sub-token without "##" prefix i.e. the token embedding of "মোজা". Then, we check if a token has "##" as the prefix. As long as a token has "##" prefix, we continue to add the embeddings of the sub-word tokens. When all the sub-word tokens of a word are traversed, we stop the process and calculate the average. In summary, we consider the average of the sub-token embeddings (average of the embeddings of মোজা, ##ফ, ##ফর) as the word embedding of "মোজাফফর".

3.2.3. Keywords Selection

After assigning importance scores to all words in a text, we use a threshold value which indicates the percentage of words to be extracted. For texts with 10 or more words, we returned the top 60% of words in an unordered manner. For texts with 5 to 9 words, we returned 70% of the words, while for texts with four words or fewer, we returned 80% of the words. There are two reasons behind such threshold values. First, texts with a smaller number of words inherently have less information available. Therefore, a higher threshold can be used to ensure that enough relevant words are captured to represent the content adequately. Second, shorter texts do not contain enough context to accurately determine the importance of individual tokens. By returning a higher percentage of words, the model compensates for the lack of contextual information and provide a broader representation of the content. However, it is possible that such threshold value on shorter texts can force the extractor to select some less important words. For example, we consider a short text in Bangla "তাই সে কাজটি করেনি" (English Translation: "That is why he did not do the work"). The Bangla text has only four words, while the corresponding translated English text has nine words. In this text, the two important words are "কাজটি" (the work) and "করেনি" (did not do). During text generation, the generation model is free to generate text using these two words with other diverse words. But including very few less important words i.e. "তাই" (that is why) will not hurt the text generation performance instead will force the generation model to generate a text that contains less important words. By lowering the threshold value, we can control the selection of less important keywords.

3.2.4. Baselines

For our proposed keyword extraction approach, we employed two publicly available pre-trained Bangla language models: BanglaBERT (Bhattacharjee et al., 2022) and Bangla-BERT-Base (Sarker, 2020). For performance comparison, we applied two other popular keyword extraction techniques: TextRank (Mihalcea and Tarau, 2004) and YAKE (Campos et al., 2020).

3.2.5. Evaluation Metrics

We utilized Mean Reciprocal Rank (MRR) (Voorhees et al., 1999), Mean Average Precision (mAP) (Baeza-Yates et al., 1999) and Normalized

Discounted Cumulative Gain (nDCG) (Järvelin and Kekäläinen, 2017) scores to compare the performance of the keyword extractors.

3.2.6. Performance Analysis

We assessed the keyword extractors using the above mentioned evaluation metrics. To make comparisons, we conducted evaluations on the initial $10K$ data points from the test dataset, which contained manually extracted keywords. The findings are presented in Table 3. It is evident

Metrics	Bangla BERT	Bangla BERT Base	Text Rank	YAKE
MRR	**33.59**	33.49	33.08	33.55
mAP	**33.56**	32.43	31.60	31.63
nDCG	33.81	31.11	27.01	**37.91**

Table 3: Comparative analysis among four keyword extraction methods. Results are expressed in percentages.

from the table that both BanglaBERT and Bangla-BERT-Base exhibited similar performance across all metrics. However, BanglaBERT attained higher scores. They achieved slightly higher Mean Reciprocal Rank (MRR) and Mean Average Precision (mAP) compared to TextRank and YAKE. TextRank consistently performed slightly lower than the BERT-based models in terms of MRR and mAP, but notably lower in terms of nDCG. This suggests that TextRank does not capture the nuances of the dataset as effectively as BERT-based methods, especially in terms of ranking relevance. YAKE outperforms other techniques significantly in terms of nDCG, indicating that it is more effective in capturing the relevance of keywords in the dataset. However, it performs lower in terms of MRR and mAP compared to BanglaBERT and Bangla-BERT-Base. Based on the results, we employed BanglaBERT based keyword extractor (as our proposed) to automatically extract keywords from all the texts within the *Bangla Key2Text* dataset.

4. Text Generation

Transformer (Vaswani et al., 2017) based pre-trained sequence-to-sequence models, when fine-tuned on natural language generation tasks, can yield exceptional results (Rothe et al., 2020). We fine-tuned two pre-trained sequence-to-sequence T5 (Raffel et al., 2020) based models for Bangla text generation task from keywords using our *Bangla Key2Text* dataset.

4.1. Baselines

To establish baseline performance on the Bangla Key2Text dataset, we fine-tune two transformer-based encoder–decoder models, BanglaT5 (Bhattacharjee et al., 2023) and (mT5) (Hasan et al., 2021), as task-specific baselines for keyword-to-text generation. We additionally evaluate several zero-shot open-source large language models (LLMs) to examine how general-purpose models perform without task-specific fine-tuning. We also evaluated zero-shot capabilities of several 4-bit quantized (due to computational resource limitations) open-source large language models. These models include Phi-3.5 mini (3.8B), mGPT (1.3B), gemma-2 (2B), Mistral (7B), Ministral (8B), Llama 3.1 (8B), Qwen-2.5 (7B), Bangla-llama-2 (7B). These models were evaluated using their default inference settings without any fine-tuning.

4.2. Experimental Settings

We fine-tuned both the T5 models for two epochs. mT5 was fine-tuned using an NVIDIA GeForce RTX 3090 24GB GPU and BanglaT5 was fine-tuned using an NVIDIA GeForce RTX 3060 12GB GPU. All models were implemented using PyTorch (Paszke et al., 2019) framework. During the fine-tuning process, the data was organized into batches, with batch size 2 for mT5 and batch size 1 for BanglaT5. We utilized a learning rate of $1e-4$, employed the *AdamW* optimizer with a linear warm up of 500 steps and 'OneCycleLR' learning rate scheduler. The scheduler was configured with various parameters including a division factor of 10, a final division factor of 100, a starting percentage of 10% and a linear annealing strategy. To accommodate the model's token limitations, we truncated the input sequences to 512 tokens and added PAD tokens where necessary. Considering the average number of keywords and the average number of words in a text in our *Bangla Key2Text* dataset, we set the max length for text generation to 64 tokens. During inference, we experimented with different decoding techniques i.e. greedy decoding, beam search with beam size 2, top-k sampling with $k = 50$ and top-p sampling with $p = 0.95$. The repetition penalty and length penalty were set to 2.5 and $\alpha = 1.0$ respectively. For 4-bit quantized LLMs, we utilized model-specific default text generation parameters during inference.

4.3. Evaluation Metrics

We assessed the performance of the text generation models using several standard natural language generation (NLG) metrics that includes BERTScore (Zhang et al., 2019), ROUGE (Lin, 2004), BLEU (Papineni et al., 2002), WER (Ali and

Metrics	Top-p		Beam		Greedy		Top-k		Top-p & Top-k	
	mT5	bnT5	mT5	bnT5	mT5	bnT5	mT5	bnT5	mT5	bnT5
BERT-S	**91.07**	91.51	90.73	92.76	89.02	92.45	91.1	92.38	91.03	92.41
ROUGE-1	**60.63**	56.34	59.6	61.89	50.84	61.36	60.04	60.96	60.6	61.15
ROUGE-L	45.18	43.23	44.02	48.66	37.61	47.62	**45.72**	47.19	45.02	47.42
BLEU-3	11.59	11.36	10.78	16.45	6.55	15.37	11.44	14.91	**12.59**	15.04
BLEU-4	5.04	5.06	4.44	8.16	2.45	7.61	5.04	7.36	**5.07**	7.46
WER	**75.3**	79.9	76.96	71.82	86.46	73.6	76.31	74.2	76.18	74.06
WIL	**83.47**	84.81	84.57	80.03	89.38	80.99	84.09	81.34	84.61	81.27

Table 4: Text generation performance comparison between the *mT5* and *BanglaT5 (bnT5)* models on initial $10K$ test data using various decoding techniques. Results are presented as percentages. BERT-S represents BERTScore. **Bold** values indicate the maximum scores achieved by the *mT5* model, while underlined values indicate the maximum scores achieved by the *BanglaT5* model.

Renals, 2018), and WIL[3]. We also conducted human evaluations on the quality of the generated texts by native Bangla speakers.

4.4. Results and Discussion

In this section, we address the performance of the text generation models based on extensive experimentation.

(a) Selection of decoding technique. We tested five decoding methods - *greedy*, *beam search*, *top-k sampling*, *top-p sampling*, and a combination of *top-p & top-k* on the $10K$ evaluation set for both $\mathrm{mT5}$ and $\mathrm{BanglaT5}$ to identify the most effective approach, then applied the best method to the full $100K$ test set. Results (Table 4) show that $\mathrm{mT5}$ performed best with *top-p sampling*, while $\mathrm{BanglaT5}$ achieved its highest scores with *beam search*.

(b) Performance of the fine-tuned models. We evaluated the proposed fine-tuned models on the $100K$ test set using the specified metrics (Table 5), applying *top-p sampling* for $\mathrm{mT5}$ and *beam search* for $\mathrm{BanglaT5}$. BERTScore (F1) reached 91.06% for $\mathrm{mT5}$ and 91.51% for $\mathrm{BanglaT5}$, reflecting strong semantic alignment between generated and reference texts. ROUGE-L was lower - 45.15% and 43.20%, respectively - indicating room for improvement in capturing longer sequences. BLEU-3 and BLEU-4 were also relatively low, showing limited higher-order n-gram similarity. Word Error Rate (WER) was high (75.34% for $\mathrm{mT5}$, 79.98% for $\mathrm{BanglaT5}$), while Word Information Lost (WIL) scores (83.47% and 84.85%) suggest notable information loss at the word level.

Metrics	mT5	BanglaT5
BERTScore	91.06	**91.51**
ROUGE-1	**60.62**	56.36
ROUGE-L	**45.15**	43.20
BLEU-3	**11.68**	11.37
BLEU-4	5.06	**5.07**
WER	**75.34**	79.98
WIL	**83.47**	84.85

Table 5: Text generation performance comparison between the *mT5* and *BanglaT5* models on the $100K$ test set. Results are presented as percentages.

We observe relatively high BERTScore values alongside high WER. This occurs because multiple valid sentences can be generated from the same keywords, leading to lexical differences from the reference text; BERTScore captures semantic similarity while WER penalizes surface-level mismatches.

(c) Impact of Temperature. We studied how the temperature parameter (τ) affects text generation in $\mathrm{mT5}$ and $\mathrm{BanglaT5}$ using the initial $10K$ evaluation set and three values ($\tau = 0.3, 0.7, 1.0$). Performance at $\tau = 1.0$ is shown in Table 4, while results for $\tau = 0.3$ and $\tau = 0.7$ are in Table 6. For $\mathrm{mT5}$, we used top-p sampling; for $\mathrm{BanglaT5}$, greedy and beam search - since these worked best at $\tau = 1.0$. The results show that for $\mathrm{BanglaT5}$, beam search with $\tau = 0.7$ outperforms $\tau = 0.3$ across all metrics, with a similar trend for greedy decoding. Likewise, in $\mathrm{mT5}$, lowering τ from 0.7 to 0.3 reduces performance.

Metrics	BanglaT5 (Beam)		BanglaT5 (Greedy)		mT5 (Top-p)	
	$\tau = 0.3$	$\tau = 0.7$	$\tau = 0.3$	$\tau = 0.7$	$\tau = 0.3$	$\tau = 0.7$
BERTScore	88.6	92.7	91.95	92.38	83.8	90.5
ROUGE-1	47.29	62.26	58.68	60.96	29.35	59.22
ROUGE-L	39.07	49.13	45.12	47.22	24.35	44.29
BLEU-3	11.93	16.66	13.14	14.97	5.62	11.09
BLEU-4	5.93	8.4	6.23	7.35	2.15	4.69
WER	136.99	70.92	77.09	74.2	110.2	75.45
WIL	83.56	79.33	83.22	81.37	91.21	83.63

Table 6: Effect of different temperature settings on the text generation quality of the fine-tuned *mT5* and *BanglaT5* models.

[3] https://pytorch.org/torcheval/stable/generated/torcheval.metrics.WordInformationLost.html

(d) Generalization performance on unseen and dialect data. To evaluate generalization, we collected 1000 Bangla news articles from Prothom Alo[4] that were not included in the training corpus. Keywords were extracted both manually and automatically using our proposed extractor, and generated texts were evaluated using the same metrics as the main test set (results in Table 7). Our findings show mT5 performs best with

Metrics	Proposed Extractor		Human Extracted	
	mT5	BanglaT5	mT5	BanglaT5
BERT-S	91.25	**91.85**	90.66	**92.14**
ROUGE-1	**63.51**	59.30	60.89	**61.93**
ROUGE-L	**50.03**	48.26	48.11	**50.75**
BLEU-3	**12.12**	10.86	9.55	**12.29**
BLEU-4	**5.44**	4.33	3.78	**5.02**
WER	**72.33**	75.70	73.79	**71**
WIL	**81.68**	83.95	83.37	**81.53**

Table 7: Text generation performance comparison between the *mT5* and *BanglaT5* models on the unseen $1K$ news data. Results are presented as percentages. BERT-S represents BERTScore.

proposed extractor, while BanglaT5 performs best with human-extracted keywords. The exact match between manual and automatic keywords is 74.10%. To assess text generalization, we collected Bangla regional dialect keywords and generated texts with our fine-tuned models. The models could produce dialectal texts from the keywords, but quality was limited which is expected since neither mT5 nor BanglaT5 was fine-tuned on dialect data. Dialects pose greater challenges than standard Bangla due to varied grammar, vocabulary, idioms, and contextual sensitivities. These factors necessitate dialect-specific data and fine-tuning, which we identify as a promising direction for future work. This dialect evaluation is exploratory and qualitative, as no standardized Bangla dialect dataset exists for keyword-to-text generation.

(e) Human Evaluation. We conducted a human evaluation with three expert annotators. They reviewed 1000 randomly selected keyword–text pairs from the test set, along with outputs from both models, using our hosted inference API. Each annotator independently assessed all 1000 texts for coherence and relevance to the input keywords, considering semantic alignment with the original text and keyword presence. They labeled each pair as *relevant*, *partially relevant*, or *irrelevant*, framing the task as a data-labeling problem with distinct classes. Inter-annotator agreement was measured using Fleiss' kappa (Fleiss, 1971), yielding scores of 0.87 for mT5 and 0.84 for BanglaT5 (Table 8), indicating substantial agreement. Although

[4] https://www.prothomalo.com/

Model	Category	Kappa Score	Average
mT5	Relevant	0.91	0.87
	Irrelevant	0.83	
BanglaT5	Relevant	0.87	0.84
	Irrelevant	0.81	

Table 8: Fleiss' kappa score for inter-annotator agreement on *relevant/partially relevant* and *irrelevant* categories.

only three annotators were involved, this setup follows common practice in NLG evaluation and the high Fleiss' κ scores indicate reliable agreement.

(f) Performance comparison with LLMs. To evaluate how well general-purpose LLMs handle Bangla keyword-to-text generation, we tested eight open-source, 4-bit quantized models (due to resource constraints) in a zero-shot setting on the $10K$ evaluation set using their default decoding settings. As shown in Table 9, BanglaT5 achieved the best overall results across all metrics, with mT5 following closely - particularly strong on semantic alignment (BERTScore). In contrast, the other models, including Phi-3.5, mGPT, Gemma, Qwen, and Mistral, performed poorly, with low ROUGE and high WER/WIL scores, reflecting outputs that were often short, disfluent, or misaligned with the keywords. Even Bangla-LLaMA-2 and LLaMA-3.1, despite their larger Bangla vocabularies and instruction tuning, offered only marginal improvements - echoing earlier findings (Kabir et al., 2023; Mahfuz et al., 2025). Note that these LLMs are evaluated in a zero-shot setting without task-specific fine-tuning; our goal is to illustrate the importance of task-specific training for low-resource languages such as Bangla. These results suggest that for low-resource languages like Bangla, task-specific fine-tuning remains crucial, as general-purpose LLMs - even when compressed - struggle to generate coherent and keyword-faithful text.

(g) Effect of number of keywords during generation. In the *Bangla Key2Text* dataset, texts average 9.381 keywords and 15.142 words. Based on these values, we set the maximum generation length to 64 tokens, though this is adjustable. Trial and error by adding and removing keywords showed that generating longer sequences (65-100 tokens) requires more keywords - typically 10-15 produce satisfactory multi-sentence outputs. This is because more keywords prompt the models to add additional context words, leading to richer texts. We also tested both models with only a single keyword and found they produced very short but complete texts.

(h) Constrained Beam Search. Constrained beam search (Hokamp and Liu, 2017) enforces the inclusion of specified tokens at each genera-

Metrics	BanglaT5	mT5	Phi-3.5	mGPT	Gemma-2B	Mistral	Ministral	LLaMA-3.1	LLaMA-2	Qwen
WER	**71.82**	76.96	96.19	103.58	105.09	104.55	107.11	99.76	125.3	101.07
WIL	**80.03**	84.57	97.86	99.56	99.69	99.83	99.58	99.26	99.6	99.66
BERTScore	**92.76**	90.73	77.41	79.43	78.57	78.16	79.03	77.78	84.11	78.61
ROUGE-1	**61.89**	59.6	11.95	4.99	4.08	1.82	5.58	5.43	7.34	3.35
ROUGE-L	**48.66**	44.02	10.31	4.37	3.62	1.7	4.6	4.76	6.52	2.93
BLEU-3	**16.45**	10.78	2.32e-03	3.58e-05	1.23e-04	2.99e-05	3.02e-04	9.70e-06	8.53e-05	1.11e-105

Table 9: Performance comparison of open-source large language models (LLMs) with BanglaT5 across various metrics. **Bold** values indicate the best scores. LLaMA-2 here represents the Bangla-llama-2 model.

Keys: [B] শীতের, কম, আগে, সম্ভাবনা	**[E]** winter, slim, before, chance
Original Text: [B] শীতের আগে সেই সম্ভাবনা কম।	**[E]** That chance is slim before winter.
mT5: [B] শীতের আগে এমন সম্ভাবনা কম।	**[E]** Such chance is slim before winter.
BanglaT5: [B] শীতের আগে আসার সম্ভাবনা কম।	**[E]** The chance of arriving before winter is slim.
Reordered Keys: [B] সম্ভাবনা, কম, আগে, শীতের	**[E]** chance, slim, before, winter
mT5: [B] শীতের আগে এমন সম্ভাবনা কম।	**[E]** Such chance is slim before winter.
BanglaT5: [B] শীতের আগে এ সম্ভাবনা কম।	**[E]** The chance is slim before winter.
Keys: [B] পেল, খবর, কোত্থেকে, বানোয়াট	**[E]** get, news, where, fake
Original Text: [B] তারা কোত্থেকে পেল এসব বানোয়াট খবর?	**[E]** Where did they get these fake news?
mT5: [B] কোত্থেকে বানোয়াট খবর পেল তাঁরা?	**[E]** Where did they get fake news?
BanglaT5: [B] কোত্থেকে বানোয়াট খবর পেল তাঁরা?	**[E]** Where did they get fake news?
Reordered Keys: [B] বানোয়াট, খবর, পেল, কোত্থেকে	**[E]** fake, news, get, where
mT5: [B] কোত্থেকে বানোয়াট খবর পেল তাঁরা?	**[E]** Where did they get fake news?
BanglaT5: [B] বানোয়াট খবর কোত্থেকে পেল সে?	**[E]** Where did he get fake news?
Keys: [B] ছেলেটা, ঢাকায়, আসে, গ্রামের	**[E]** boy, Dhaka, comes, village
Original Text: [B] গ্রামের সহজ-সরল ছেলেটা ঢাকায় আসে।	**[E]** The simple village boy comes to Dhaka.
mT5: [B] ছেলেটা গ্রামের বাড়ি থেকে ঢাকায় আসে।	**[E]** The boy comes to Dhaka from his village home.
BanglaT5: [B] গ্রামের ছেলেটা ঢাকায় আসে না।	**[E]** The village boy does not come to Dhaka.
Reordered Keys: [B] গ্রামের, ছেলেটা, আসে, ঢাকায়	**[E]** village, boy, comes, Dhaka
mT5: [B] ছেলেটা গ্রামের বাড়ি থেকে ঢাকায় আসে।	**[E]** The boy comes to Dhaka from his village home.
BanglaT5: [B] গ্রামের ছেলেটা ঢাকায় আসে না।	**[E]** The village boy does not come to Dhaka.

Table 10: Few examples of generated texts by the fine-tuned *mT5* and *BanglaT5* models, showcasing the impact of different input keyword orders. Each row represents a distinct example, with two variations of keyword order provided. These examples are drawn from the test dataset. **B** represents the keywords or texts in Bangla and **E** represents their corresponding English translation.

tion step. In our experiments, it successfully ensured missing keywords appeared in the output but had limitations. We used a two-stage brute-force approach: first generating text with standard beam search, then checking which keywords were missing, and finally regenerating the text with constrained beam search to enforce their inclusion while keeping other parameters unchanged. However, forcing all keywords in this way reduced text quality, as compelling the model to include one keyword often led to the exclusion of others.

(i) Cross-Lingual Transfer. During generation, we observed cross-lingual transfer (Schuster et al., 2019), where both $\mathrm{mT5}$ and $\mathrm{BanglaT5}$ produced Bangla text even when given mixed Bangla-English keywords. Fine-tuned T5 models specialize in their training language, while multilingual models like $\mathrm{mT5}$ leverage cross-lingual understanding. Since both models were fine-tuned on Bangla, they favor Bangla outputs. For $\mathrm{mT5}$, this means interpreting English keywords, mapping them to Bangla, and generating consistent Bangla text. $\mathrm{BanglaT5}$, however, sometimes retains English words in the output, though their placement remains semantically appropriate. Future research could further explore this cross-lingual behavior and develop better strategies to control mixed generation.

(j) Sensitivity to Keyword Order We conducted manual testing on the text generation behavior of

the fine-tuned models. We randomly selected few keyword-text pairs from the test dataset. For each pair, we shuffled the keyword order (both randomly and preserving the original appearance order in the input text) and observed the generated texts from both models. We present some examples in Table 10. Our focus was on assessing the impact of using different variations of input keyword order on the text generation process. The examples illustrate that altering the input keyword order results in minor differences in the generated output, highlighting the model's sensitivity to keyword order. However, in all the examples, although the syntactic structures of the generated texts differ, their semantic meaning remains consistent compared to the original text. In most cases, even with changes in keyword order, the generated text maintains both syntactic and semantic coherence. This observation aligns with the high BERTScores 91.06% ($\mathrm{mT5}$) and 91.51% ($\mathrm{BanglaT5}$) indicating substantial semantic correctness between the reference and generated texts. Nevertheless, it is essential to quantify the sensitivity of the text generation models to changes in keyword order using proper evaluation metrics and delve deeper into analyzing the effects. We leave this task open for future work.

5. Conclusion and Future Works

In this work, we introduced *Bangla Key2Text*, a large-scale dataset of 2.6 million Bangla keyword-text pairs designed for keyword-to-text generation. We developed a BERT-based keyword extractor and fine-tuned $\mathrm{mT5}$ and $\mathrm{BanglaT5}$, demonstrating through extensive evaluation that they generate coherent, keyword-faithful Bangla text. Our experiments - covering zero-shot comparisons with multiple compressed LLMs, human evaluations, unseen data, and dialectal inputs - show that task-specific fine-tuning of smaller language models significantly outperforms general-purpose models in this low-resource setting. Moving forward, we plan to fine-tune LLMs and monolingual models (Bhattacharjee et al., 2023), explore few-shot learning, and further investigate cross-lingual keyword-to-text generation (Chi et al., 2020).

6. Limitations

We observed that the generated texts occasionally failed to incorporate all provided keywords, leading to incomplete representation of the intended content. This highlights the need for more effective decoding strategies, such as improved constrained generation. Also, most extracted keywords used for fine-tuning were not lemmatized, which sometimes caused fluency and grammatical issues when generating text from lemmatized or morphologically varied keywords. Additionally, since the dataset is derived primarily from news articles, models trained on it may generalize less effectively to conversational or informal Bangla. Another limitation is that our models lack mechanisms to detect or filter harmful content such as slang, hate speech, or adult text, raising ethical and safety concerns. Moreover, while our fine-tuned models significantly outperform general-purpose LLMs, their performance on dialectal and cross-lingual inputs remains limited, and the cross-lingual transfer behavior observed warrants deeper investigation. Finally, as our work focused on 4-bit quantized and fine-tuned models, the impact of higher-precision models or more extensive fine-tuning remains unexplored. Addressing these limitations could improve the reliability, safety, and generalizability of Bangla keyword-to-text generation.

7. Ethical Considerations

Our dataset, Bangla Key2Text, was compiled exclusively from publicly available Bangla news articles and online text sources. All materials were collected and used under fair-use provisions for research and educational purposes. No personally identifiable information (PII) or user-generated private data were included. The dataset and models are intended strictly for academic and non-commercial research under open licensing guidelines.

References

Ahmed Ali and Steve Renals. 2018. Word error rate estimation for speech recognition: e-wer. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 20–24.

Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. 1999. *Modern information retrieval*, volume 463. ACM press New York.

Abhik Bhattacharjee, Tahmid Hasan, Wasi Ahmad, Kazi Samin Mubasshir, Md Saiful Islam, Anindya Iqbal, M. Sohel Rahman, and Rifat Shahriyar. 2022. Banglabert: Language model pretraining and benchmarks for low-resource language understanding evaluation in bangla. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1318–1327, Seattle, United States. Association for Computational Linguistics.

Abhik Bhattacharjee, Tahmid Hasan, Wasi Uddin Ahmad, and Rifat Shahriyar. 2023. Banglanlg

and banglat5: Benchmarks and resources for evaluating low-resource natural language generation in bangla. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 726–735, Dubrovnik, Croatia. Association for Computational Linguistics.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

Zewen Chi, Li Dong, Furu Wei, Wenhui Wang, Xian-Ling Mao, and Heyan Huang. 2020. Cross-lingual natural language generation via pre-training. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7570–7577.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. 2020. A survey of multilingual neural machine translation. *ACM Computing Surveys (CSUR)*, 53(5):1–38.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Rahhal Errattahi, Asmaa El Hannani, and Hassan Ouahmane. 2018. Automatic speech recognition errors detection and correction: A review. *Procedia Computer Science*, 128:32–37.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Cristina Garbacea and Qiaozhu Mei. 2020. Neural language generation: Formulation, methods, and evaluation. *arXiv preprint arXiv:2007.15780*.

Fan Ge and Li Kuang. 2021. Keywords guided method name generation. In *2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC)*, pages 196–206. IEEE.

Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. 2021. Xl-sum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703, Online. Association for Computational Linguistics.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Kalervo Järvelin and Jaana Kekäläinen. 2017. Ir evaluation methods for retrieving highly relevant documents. In *ACM SIGIR Forum*, volume 51, pages 243–250. ACM New York, NY, USA.

Mohsinul Kabir, Mohammed Saidul Islam, Md Tahmid Rahman Laskar, Mir Tafseer Nayeem, M Saiful Bari, and Enamul Hoque. 2023. Benllmeval: A comprehensive evaluation into the potentials and pitfalls of large language models on bengali nlp. *arXiv preprint arXiv:2309.13173*.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436–444.

Haoran Li, Junnan Zhu, Jiajun Zhang, Chengqing Zong, and Xiaodong He. 2020. Keywords-guided abstractive sentence summarization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8196–8203.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Tamzeed Mahfuz, Satak Kumar Dey, Ruwad Naswan, Hasnaen Adil, Khondker Salman Sayeed, and Haz Sameen Shahgir. 2025. Too late to train, too early to use? a study on necessity and viability of low-resource Bengali LLMs. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 1183–1200, Abu Dhabi, UAE. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Abhijit Mishra, Md Faisal Mahbub Chowdhury, Sagar Manohar, Dan Gutfreund, and Karthik Sankaranarayanan. 2020. Template controllable keywords-to-text generation. *arXiv preprint arXiv:2011.03722*.

Zabir Al Nazi. 2020. Bangla newspaper dataset.

Jinran Nie, Liner Yang, Yun Chen, Cunliang Kong, Junhui Zhu, and Erhong Yang. 2022. Lexical complexity controlled sentence generation. *arXiv preprint arXiv:2211.14540*.

Gözde Özbal, Daniele Pighin, and Carlo Strapparava. 2013. Brainsup: Brainstorming support for creative sentence generation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1446–1455.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Dongju Park and Chang Wook Ahn. 2018. Lstm encoder-decoder with adversarial network for text generation from keyword. In *Bio-inspired Computing: Theories and Applications: 13th International Conference, BIC-TA 2018, Beijing, China, November 2–4, 2018, Proceedings, Part II 13*, pages 388–396. Springer.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Ilan Price, Jordan Gifford-Moore, Jory Fleming, Saul Musker, Maayan Roichman, Guillaume Sylvain, Nithum Thain, Lucas Dixon, and Jeffrey Sorensen. 2020. Six attributes of unhealthy conversation. *arXiv preprint arXiv:2010.07410*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Sagor Sarker. 2020. Banglabert: Bengali mask language model for bengali language understanding.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805, Minneapolis, Minnesota. Association for Computational Linguistics.

Junxiao Shen, Boyin Yang, John J Dudley, and Per Ola Kristensson. 2022. Kwickchat: A multi-turn dialogue system for aac using context-aware sentence generation by bag-of-keywords. In *27th International Conference on Intelligent User Interfaces*, pages 853–867.

Sifatullah Siddiqi and Aditi Sharan. 2015. Keyword and keyphrase extraction techniques: a literature review. *International Journal of Computer Applications*, 109(2).

Zhenqiao Song, Xiaoqing Zheng, Lu Liu, Mu Xu, and Xuan-Jing Huang. 2019a. Generating responses with a specific emotion in dialog. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3695.

Ziyao Song, Lizhen Liu, Wei Song, Xinlei Zhao, and Chao Du. 2019b. A neural network model for chinese sentence generation with key word. In *2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, pages 334–337. IEEE.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Kiyotaka Uchimoto, Satoshi Sekine, and Hitoshi Isahara. 2002. Text generation from keywords. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ellen M Voorhees et al. 1999. The trec-8 question answering track report. In *Trec*, volume 99, pages 77–82.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Cheng Yong, Mao Yingchi, Wang Yi, Ping Ping, and Wang Longbao. 2021. Keywords-based dam defect image caption generation. In *2021 IEEE Seventh International Conference on Big Data Computing Service and Applications (BigDataService)*, pages 214–221. IEEE.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Appendix

A. Experimental Settings

A.1. Keyword Extraction

The BERT based models were utilized for inference using their default hyper-parameters in the keyword extraction phase. No fine-tuning was performed. The tokenized text is passed through the pre-trained model to obtain the *last hidden state* as the token embeddings, which had a shape of 512×768. By computing the average of the embeddings across the 512 tokens, we derived a mean embedding of shape 768.

A.2. Text Generation

We fine-tuned both the T5 models for two epochs. $\mathrm{mT5}$ was fine-tuned using an NVIDIA GeForce RTX 3090 24GB GPU and $\mathrm{BanglaT5}$ was fine-tuned using an NVIDIA GeForce RTX 3060 12GB GPU. All models were implemented using $\mathrm{PyTorch}$ (Paszke et al., 2019) framework. During the fine-tuning process, the data was organized into batches, with batch size 2 for $\mathrm{mT5}$ and batch size 1 for $\mathrm{BanglaT5}$. We utilized a learning rate of $1e-4$, employed the *AdamW* optimizer with a linear warm up of 500 steps and 'OneCycleLR' learning rate scheduler. The scheduler was configured with various parameters including a division factor of 10, a final division factor of 100, a starting percentage of 10% and a linear annealing strategy. To accommodate the model's token limitations, we truncated the input sequences to 512 tokens and added PAD tokens where necessary. Considering the average number of keywords and the average number of words in a text in our *Bangla Key2Text* dataset, we set the max length for text generation to 64 tokens. During inference, we experimented with different decoding techniques i.e. greedy decoding, beam search with beam size 2, top-k sampling with $k = 50$ and top-p sampling with $p = 0.95$. The repetition penalty and length penalty were set to 2.5 and $\alpha = 1.0$ respectively. For 4-bit quantized LLMs, we utilized model-specific default text generation parameters during inference.

B. Model Descriptions

B.1. Keywords Extraction

(a) BanglaBERT. A BERT model (Bhattacharjee et al., 2022) fine-tuned for performing Bangla downstream tasks. The model checkpoint[5] can be accessed through the Hugging Face Transformers Library (Wolf et al., 2020). This discriminator model is pre-trained using ELECTRA (Clark et al., 2020) approach. It is comprised of a total of 110 million parameters, with 12 hidden layers and 12 attention heads. The embedding size of the model is 768 and the tokenizer associated with it has a vocabulary size of $32,000$.

(b) Bangla-BERT-Base. Another BERT model (Sarker, 2020) fine-tuned for performing Bangla downstream tasks. We fine-tuned the model checkpoint[6] available on Hugging Face. It has the same embedding size as well as the same number of parameters, hidden layers, attention heads as the $\mathrm{BanglaBERT}$ model. However, the tokenizer associated with it has a vocabulary size of $102,025$.

(c) TextRank. An algorithm (Mihalcea and Tarau, 2004) for text processing that is based on the $\mathrm{PageRank}$ (Page et al., 1999) algorithm used by Google for ranking web pages and websites. For keyword extraction, $\mathrm{TextRank}$ identifies words in the text that are central and appear more often in the context of other words. The main idea is that words or phrases in a text are represented as nodes in a graph, and the connections between

[5] https://huggingface.co/csebuetnlp/banglabert

[6] https://huggingface.co/sagorsarker/bangla-bert-base

them are based on their co-occurrence in a certain window of words or based on other similarity measures. It works by assigning a score to each node in the graph based on the principle that a node is important if it is linked to by other important nodes. The algorithm iteratively calculates these scores until the process converges, meaning that the scores do not change significantly with further iterations. We employed a pre-trained BanglaBERT model checkpoint[7] from Hugging Face to compute the edge weights through cosine distance measurements among the text embeddings.

(d) YAKE. **Y**et **A**nother **K**eyword **E**xtractor (Campos et al., 2020) operates as an unsupervised automatic keyword extraction technique, relying on statistical features extracted from individual documents to identify the most significant keywords. The system does not require training on specific documents, nor does it rely on dictionaries or external corpora. It is publicly available as a Python library[8].

B.2. Text Generation

(a) mT5. A multilingual T5 model (Hasan et al., 2021) fine-tuned extensively with abstractive summarization data covering 44 different languages. We trained the model checkpoint[9] available on the Hugging Face Transformers Library (Wolf et al., 2020) which is a fine-tuned version of the base mT5 model (**?**). It contains a total of 600 million parameters and consists of 12 layers in both the encoder and decoder components. Each layer is equipped with 12 attention heads. The tokenizer associated with this model has a vocabulary size of $250,112$.

(b) BanglaT5. A sequence-to-sequence T5 (Bhattacharjee et al., 2023) model that has been pre-trained using the span corruption objective with extensive Bangla texts to perform Bangla NLG tasks. The pre-trained model checkpoint[10] is available on the Hugging Face Library. It has the same number of encoder and decoder layers as the mT5. However, it consists of 247 million parameters and has a vocabulary of size $32,128$.

(c) Large Language Models. We additionally evaluated eight open-source, 4-bit quantized large language models in a zero-shot setting on the $10K$ evaluation set. These included Phi-3.5-mini-instruct[11] (3.8B parameters), mGPT[12] (1.3B parameters), Gemma-2B[13] (2B parameters), Mistral-7B[14] (7B parameters), Ministral-8B[15] (8B parameters), LLaMA-3.1–8B[16] (8B parameters), Qwen2.5-7B[17] (7B parameters), and Bangla-LLaMA-2-7B[18] (7B parameters). All models were used with their publicly available instruction-tuned checkpoints from Hugging Face and evaluated using their default decoding settings without additional fine-tuning.

C. Performance Evaluation Metrics

C.1. Keyword Extraction

MRR. **M**ean **R**eciprocal **R**ank (Voorhees et al., 1999) assesses the average inverse position of the first accurately identified keyword within a group of queries. For each query, the reciprocal rank represents the multiplicative inverse of the position of the initial relevant keyword, and MRR is the mean of these values across all queries. A higher MRR indicates superior performance of the keyword extraction tool.

mAP. **M**ean **A**verage **P**recision (Baeza-Yates et al., 1999) evaluates the average precision of outcomes at every position in the ordered list. Precision denotes the proportion of pertinent results within the top returned outcomes. mAP is computed by averaging the precision scores for each position in the ordered list. Enhanced performance of the keyword extraction tool is indicated by a higher mAP.

nDCG. **N**ormalized **D**iscounted **C**umulative **G**ain (Järvelin and Kekäläinen, 2017) evaluates the ranking quality of keyword extraction tools by taking into account the position of each relevant keyword, giving higher importance to keywords ranked higher on the list. It compares the relevance of the extracted keywords with an ideal list of keywords, and discounts the relevance of each keyword based on its position in the ranked list. The

[7] https://huggingface.co/csebuetnlp/banglabert_generator

[8] https://github.com/LIAAD/yake

[9] https://huggingface.co/csebuetnlp/mT5_multilingual_XLSum

[10] https://huggingface.co/csebuetnlp/banglat5

[11] https://huggingface.co/microsoft/Phi-3.5-mini-instruct

[12] https://huggingface.co/ai-forever/mGPT

[13] https://huggingface.co/google/gemma-2-2b

[14] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3

[15] https://huggingface.co/mistralai/Ministral-8B-Instruct-2410

[16] https://huggingface.co/meta-llama/Meta-Llama-3.1-8B-Instruct

[17] https://huggingface.co/Qwen/Qwen2.5-7B-Instruct

[18] https://huggingface.co/BanglaLLM/bangla-llama-7b-instruct-v0.1

cumulative gain is then normalized against the ideal ranking to yield a score between 0 and 1, where 1 indicates the perfect ranking of keywords.

C.2. Text Generation

BLEU. **Bi** **L**ingual **E**valuation **U**nderstudy (**?**) assesses the similarity between reference and candidate texts by examining the overlap of their n-grams. It evaluates the number of matches between the generated and reference texts across different n-gram lengths (usually 1 to 4), and these matches are considered regardless of their position. The precision of these n-gram matches is calculated, and the values are aggregated using a geometric mean.

ROUGE. **R**ecall-**O**riented **U**nderstudy for **G**isting **E**valuation (**?**) evaluates the quality of a generated text by comparing it to a reference text, assessing the overlap of content. ROUGE-N specifically gauges the overlap of N-grams between the generated and reference texts. For instance, ROUGE-1 evaluates the match of unigrams (individual words), while ROUGE-2 examines bi-gram overlap (pairs of words), and so forth. On the other hand, ROUGE-L assesses the longest common sub-sequence (LCS) between the generated and reference texts, capturing sentence-level structural similarities and automatically identifying the longest co-occurring in-sequence n-grams. In this study, we utilized ROUGE-1 and ROUGE-L metrics in terms of F-measure, which is the harmonic mean of precision and recall.

BERTScore. BERTScore (Zhang et al., 2019) employs the cosine similarity of contextual embeddings from a BERT-based model. The fundamental concept of BERTScore lies in contrasting the contextual embeddings of tokens within the generated text with those in the reference text, thereby considering the context of each word in the comparison. In this study, we used the bert-score[19] library, which uses a multilingual BERT for Bangla texts.

WER. **W**ord **E**rror **R**ate (Errattahi et al., 2018) evaluates the error rate by contrasting a generated text with a reference text known to be accurate. WER is computed using the Levenshtein distance, which tallies the number of single-character edits (insertions, deletions, or substitutions) needed to transform one word sequence into another. A lower WER denotes superior performance, with a perfect score of 0% indicating exact correspondence between the generated and the reference text. Conversely, a higher WER signifies greater disparities between the generated and the reference text, indicating inferior performance.

WIL. **W**ord **I**nformation **L**ost (Errattahi et al., 2018) is a metric similar to Word Error Rate (WER). While WER straightforwardly calculates errors based on insertions, deletions, and substitutions, WIL offers a distinct perspective by evaluating the amount of information lost or misrepresented during the text generation process. WIL is rooted in information theory and considers the probability of words as they occur in the language. The objective is to gauge errors by the informational content of the involved words, thereby assigning greater severity to errors on less predictable (more informative) words compared to more predictable ones. The logarithm of the probability, often represented as the base-2 logarithm, effectively quantifies the information content of the word. Consequently, words with higher predictability (and thus lower probability) convey less information. Consequently, WIL penalizes errors on less predictable words more significantly. Similar to WER, WIL is expressed as a percentage, with lower values indicating superior performance.

D. Potential Applications of the Dataset

Having a dataset comprising 2.6 million Bangla keyword-text pairs presents numerous opportunities for downstream tasks in Bangla Natural Language Processing (NLP) and Natural Language Generation (NLG). This dataset can be employed to evaluate keyword extraction tools, enabling the generation of concise summaries from large documents or text passages based on specified keywords. It will be interesting to extend the dataset to support other low-resource languages through multilingual machine translation techniques (Dabre et al., 2020). Additionally, it can facilitate document classification tasks where the goal is to classify documents into predefined categories or topics based on their content (i.e. scientific articles into predefined categories). Keywords can serve as features for classification. Furthermore, the dataset can support information retrieval tasks by aiding in the retrieval of relevant documents or text passages based on user queries or keywords. Moreover, it can be utilized for training Named Entity Recognition (NER) models to identify and classify named entities mentioned in the text such as names of people, organizations, locations, dates, etc. With such a large dataset, question-answering and question-generation models can be trained to provide responses and generate questions accurately based on the provided keywords and contextual information from the text. Furthermore, text classifica-

[19] https://pypi.org/project/bert-score/

tion models can be trained to categorize text documents or passages into different classes or labels based on the provided keywords, enabling applications such as topic classification.

E. Human Annotation

E.1. Annotator Recruitment

All the annotators involved in our study are native speakers of Bangla and possess experience in NLP. It is important to note that we assigned different annotators for keyword extraction (three) and evaluating generated texts (three). This means that annotators responsible for keyword extraction did not participate in evaluating text quality. For both tasks, we selected annotators based on their trustworthiness score (Price et al., 2020), which was determined through separate tests. Specifically, to recruit annotators for evaluating the quality of generated text, we provided them with 100 keyword-text pairs and asked if the text fully contained the given keywords. From the test dataset, we randomly selected 80 keywords-text pairs, while the remaining 20 pairs served as control samples. The control samples consisted of *keywords-generated text* pairs that were unknown to the participants beforehand. After completing the task, we analyzed the number of correctly labeled control samples for each annotator. A similar procedure was followed to recruit annotators for the keyword extraction task, where annotators were asked to extract important keywords from a text. Only annotators who achieved a trustworthiness score above 90% were selected.

E.2. Annotator Agreement

Kappa Score. A statistical measure specifically designed for assessing agreement among multiple annotators. The Fleiss' kappa score (Fleiss, 1971) is an extension of Cohen's kappa coefficient (Cohen, 1960) for situations where there are multiple annotators involved. The value of Fleiss' kappa can range from -1 to 1. A value of 1 signifies perfect agreement, while 0 suggests that the annotator' judgments are independent of each other. Negative values indicate significant disagreement among the annotators.

F. Few Experimental Result Tables for Text Generation

Keys: [B] অকাইঠারে, কৈয়া, কতা, নাই	**[E]** idle, saying, words, no
Original Text: [B] অকাইঠারে বালা কতা কৈয়া লাব নাই।	**[E]** There is no use in saying good words to the idle.
mT5: [B] অকাইঠারে কৈয়া কতা নাই।	**[E]** There is nothing to say to the idle.
BanglaT5: [B] অকাইঠারে কতা কইয়া লাভ নাই।	**[E]** There is no use in talking to the idle.
Region: Mymensingh	
Keys: [B] আমগো, নাম, পুরান, হুনছেন, ঢাকার	**[E]** Our, name, old, hear, Dhaka
Original Text: [B] আমগো পুরান ঢাকার ছক্কু বেপারী লেনের নাম হুনছেন!	**[E]** Have you heard the name of the Sakku Bepari lane of our old Dhaka?
mT5: [B] পুরান ঢাকার আমগো নাম হুনছেন 'মৌসুমী'।	**[E]** 'Mousumi' has heard the name of our old Dhaka.
BanglaT5: [B] আমগো পুরান ঢাকার নাম হুনছেন না?	**[E]** Haven't you heard the name of our old Dhaka?
Region: Dhaka	
kl**Keys: [B]** কইলেই, লম্বর, নাম, পাইলে, দিবো, আমার, দেখায়	**[E]** say, number, name, find, my, show
Original Text: [B] লম্বর খুইজা না পাইলে আমার নাম কইলেই মাইনছে দেখায় দিবো।	**[E]** If you can't find the number, then people will show you when you say my name.
mT5: [B] লম্বর কইলেই আমার নাম দেখায়, পাইলে আমি দিবো না।	**[E]** Saying the number shows my name; I won't give it if you find it.
BanglaT5: [B] আমার নাম কইলেই লম্বর দেখায়, আমি টাকা দিবো।	**[E]** Saying my name shows the number, I will give the money.
Region: Dhaka	
Keys: [B] কড়ির, ভাগ, দেও, মোর	**[E]** money, share, give, my
Original Text: [B] মোর পইসা কড়ির ভাগ মোক দেও।	**[E]** Give me my share of the money.
mT5: [B] মোর কড়ির ভাগ দেও না।	[E] Give share of my money.
BanglaT5: [B] মোর কড়ির ভাগ দেও, না হয়।	[E] At least give me my share of the money.
Region: Barisal	

Table 11: The text generation capabilities of the fine-tuned *mT5* and *BanglaT5* models on dialectal data. Each row represent one data. **Region** denotes the geographical source of the dialectal data. **B** represents the keywords or texts in Bangla and **E** represents their corresponding English translation.

Keys: [B] patient, সপ্তাহ, বাড়ছে, হাসপাতালে, গত	**[E]** patient, week, increasing, hospital, last
mT5: [B] গত সপ্তাহ থেকে হাসপাতালে রোগীর সংখ্যা বাড়ছে।	**[E]** The number of patients in the hospital has been increasing since last week.
BanglaT5: [B] গত এক সপ্তাহ ধরে হাসপাতালে রোগীর সংখ্যা বাড়ছে।	**[E]** The number of patients in the hospital has been increasing for the last one week.
Keys: [B] গ্রামের, teacher, একমাত্র	**[E]** village, teacher, only
mT5: [B] গ্রামের একমাত্র শিক্ষক।	**[E]** The only teacher in the village.
BanglaT5: [B] গ্রামের একমাত্র শিক্ষক তিনি।	**[E]** He is the only teacher in the village.
Keys: [B] দেরিতে, মৃত্যু, doctor, রোগীর	**[E]** delay, died, doctor, patient
mT5: [B] দেরিতে থেকেই চিকিৎসাধীন রোগীর মৃত্যু হয়েছে বলে জানিয়েছেন হাসপাতালের একজন ডাক্তার।	**[E]** A doctor of the hospital said that the patient under treatment died due to delay.
BanglaT5: [B] Doctor: রোগীর মৃত্যু দেরিতে হয়।	**[E]** Doctor: Patient dies due to delay.
Keys: [B] প্রশ্নগুলোর, আজ, answer	**[E]** questions, today, answer
mT5: [B] আজ আপনার প্রশ্নগুলোর উত্তর পাবেন।	**[E]** You will get the answers to your questions today.
BanglaT5: [B] আজ প্রশ্নগুলোর Answer দেবেন।	**[E]** Answer the questions today.
Keys: village, teacher, only	
mT5: [B] আমিও একজন স্কুলশিক্ষক, শিক্ষক-শিক্ষার্থীরই (বিদ্যালয়)।	**[E]** I am also a school teacher, (school) is for teachers and students.
BanglaT5: [B] শুধু একটা স্কুলশিক্ষক গ্রামে।	**[E]** There is only a school teacher in the village.
Keys: increase, hospital, last	
mT5: [B] এর আগে হাসপাতালেই মৃত্যুর সংখ্যা increase।	**[E]** Before this, the number of deaths in the hospital increased.
BanglaT5: [B] সর্বশেষ গত বছর hospitalize সূচক বেড়েছে।	**[E]**The hospitalize index increased last year.
Keys: questions, today, answer	
mT5: [B] কিন্তু আপনার প্রশ্নগুলোও আজকালই 'উত্তর'।	**[E]** But your questions are also 'answers' nowadays.
BanglaT5: [B] আজকে প্রশ্নের উত্তর দাও।	**[E]** Give answers to the questions today.
Keys: come, police	
mT5: [B] পুলিশও আছে।	**[E]** There are also police.
BanglaT5: [B] police এসে যাবে।	**[E]** The police will come.

Table 12: Few instances demonstrating the cross-lingual transfer ability in the generated texts produced by both the fine-tuned models (*mT5* and *BanglaT5*) when provided with cross-lingual keywords. **B** represents the keywords or texts in Bangla and **E** represents their corresponding English translation.

Keyword	mT5 Text	BanglaT5 Text
গতকাল (yesterday)	গতকাল শুক্রবার। (Yesterday was Friday.)	গতকাল শনিবার। (Yesterday was Saturday.)
আসবে (will come)	আসবে কীভাবে? (How come?)	পুলিশ আসবে। (The police will come.)
এবার (now)	এবার কী? (What now?)	এবার ব্যতিক্রম। (Now is an exception.)
নববর্ষ (New Year)	নববর্ষ শুরু। (New Year begins.)	শুভ নববর্ষ। (Happy New Year.)

Table 13: Few examples of generated texts by both fine-tuned models with a single keyword.

Keys: [B] সঙ্গে, হয়তো, সন্ধ্যার, স্বাভাবিকভাবে, মেয়র, বহু, বিষয়ে, বলেন, মধ্যে, পাওয়া, আজ, জবাব	**[E]** with, may be, evening, naturally, Mayor, many, regard, said, by, available, today, answer
mT5: [B] আজ সন্ধ্যার মধ্যে হয়তো স্বাভাবিকভাবে জবাব পাওয়া যাবে। এ বিষয়ে আরও বলেন, মেয়র মোশাররফ হোসেনের সঙ্গে বহু দিন ধরে আলোচনা চলছে।	**[E]** The answer may be available naturally by evening today. In this regard, he further said that discussions are going on with Mayor Mosharraf Hossain for many days.
BanglaT5: [B] আজ সন্ধ্যায় মেয়র প্রার্থীদের সঙ্গে আলোচনার বিষয়টি নিশ্চিত করে আরও বলেন, তাঁদের স্বাভাবিক প্রতিক্রিয়া হয়তো অনেক ক্ষেত্রেই পাওয়া যায়। এ বিষয়ে মেয়রের ভাষ্য ছিল অনেকেরই কোনো জবাব নেই!	**[E]** Confirming the discussion with the Mayoral candidates today evening, they said their natural response may be available in many cases. The mayor's comment in this regard was that many people have no answer!
Keys: [B] মোটরসাইকেলে, শিক্ষক, ছাত্রীর, থাকা, এক, চালকের, মোড়ে, কয়েকজন, গিয়ে, ও, বাসে, একটি, তাঁদের	**[E]** on the motorcycle, teacher, of the student, staying, one, of the driver, at the corner, a few, went, and, on the bus, one, theirs
mT5: [B] এরপর কয়েকজন শিক্ষক মোড়ে গিয়ে তাঁদের বাসে ও একটি মোটরসাইকেলে আগুন দেন। পুলিশ জানায়, ঘটনাস্থলে থাকা এক ছাত্রীর চালকের সঙ্গে কথা-কাটাকাটি হয়।	**[E]** Afterwards, a few teachers went to the corner on the bus and the motorcycle, and set fire. Police informed, some conversation happened between a student of the driver and the teachers.
BanglaT5: [B] তাঁদের মধ্যে কয়েকজন শিক্ষক ও চালকের সহযোগিতায় একটি বাসে গিয়ে মোটরসাইকেলে থাকা এক ছাত্রীর লাশ উদ্ধার করে ধানমন্ডি মোড়ে পাঠানো হয়।	**[E]** Among them, with the cooperation of a few teachers and the driver, a body of a student staying on the motorcycle and the bus was retrieved from Dhanmondi corner and sent.

Table 14: Few instances demonstrating the both fine-tuned models (*mT5* and *BanglaT5*) generate texts spanning across multiple sentences. **B** represents the keywords or texts in Bangla and **E** represents their corresponding English translation.