A Hybrid Approach for Semantic Similarity of Long Bangla Texts Using BERT and Custom Transformers

Hamayath Hussain Sadi
Dept. of CSE
North East University Bangladesh
Sylhet, Bangladesh
hamayath.sadi@gmail.com

Razorshi Prozzwal Talukder
Dept. of CSE
North East University Bangladesh
Sylhet, Bangladesh
razorshi.p.t@gmail.com

***Abstract*—Semantic text similarity estimation between two long Bangla texts is a critical task issue in the field of NLP. Current similarity measuring methods such as Bert is not quite reasonable for the long texts' similarity due to the limitation of tokens(512). Due to that limitation of Bert, it cannot capture fully semantic information from the composite complex structure of long texts, turning the loss of information into a loss of accuracy in the similarity score. In this paper, we have proposed a method that is a combination of our custom transformer encoder and Bert that can directly contribute to the better accuracy of similarity scores between two long Bangla texts. Preliminary results demonstrate that our proposed method is more accurate in the Bangla long texts semantic similarity measurement task, achieving an F1 Score of 0.9939 and a Test Loss of 0.0297 during the evaluation stage, compared to the only Bert approach with common evaluation data. These evaluation insights highlight that our proposed similarity calculation framework serves as a dominant power in semantic similarity calculation tasks.**

***Index Terms*—Semantic Similarity, Bert, Custom transformer encoder, Semantic vector representation**

I. Introduction

Semantic similarity between two long Bengali texts, such as paragraphs, involves measuring how closely related the meanings of the texts are. Theoretically, semantic similarity refers to the degree of shared characteristics between two words or concepts in a language. While it is inherently a relational property between concepts or senses, it can also be described as a measure of conceptual likeness between two words, sentences, paragraphs, documents, or even larger text segments. Semantic similarity among concepts is a quantitative evaluation of information, determined based on the characteristics of the concepts and their interrelations. These similarity measures have various applications, including information extraction (IE) [1], information retrieval (IR) [2], and question-answer evaluation[3].

Text similarity calculation is a crucial task in Natural Language Processing. It is more crucial for existing low-resource languages such as Bangla. There has many research has been conducted on this semantic similarity feature [4]. Still, most works are done for Short text semantic similarity [5,6] rather than long text with multiple sentences. In 2018, google introduced Bert which gives a major boost to NLP tasks. Bert comes with contextual word embeddings which give each word a contextual numerical representation from his pre-trained embedding table [7]. Using that now we can do many specific NLP tasks with a contextual meaning, like if we give a sentence as input and extract the output from its last hidden state then we will get a sentence vector. Now this vector represents that sentence numerically with contextual meaning. So there have been many research works based on semantic similarity between two long texts in different languages [8] but unfortunately, We didn't find any work in this specific field for the Bengali language. Using Bert's embedding of two sentences we can calculate the semantic similarity by existing various methods like cosine similarity, by that we will get more richer, more accurate semantic similarity score. But there are drawbacks to this method, this method has not been ideal in the application of Bengali long text, mainly because the composition of the structure of long text with multiple sentences is more complex than short text or a single sentence and Bert only process 512 tokens after that length it takes to cut off the remaining based on the importance. By that, it creates chances to cut off semantically important information[9]. For example, Consider a Bengali news article discussing the historical and cultural significance of Durga Puja in Bangladesh. This article may contain detailed descriptions of the rituals, traditions, and social impact of the festival, exceeding the token limit of BERT. Directly applying BERT to such a text may result in the truncation of crucial information, perhaps about the economic impact of the festival, leading to an incomplete understanding of the text's overall meaning. This truncation can significantly affect the accuracy of semantic similarity calculations. For instance, if we compare the Durga Puja article to another text about the economic impact of festivals in Bangladesh, the existing BERT-based methods may fail to capture the strong semantic similarity between these two texts due to the loss of information about the economic aspects of Durga Puja. So the previously existing method can't capture the semantic information and the relationship between them throughout the complex structure of the whole long text, causing some issues with these methods' accuracy and credibility[10]. The complex structure of a long text made that task more difficult and finding the way to get rid of that makes it more urgent.

To overcome the challenge of capturing the full semantic context of long Bengali texts, this research proposes a method that utilizes sentence embeddings rather than relying solely on word embeddings. In a long text discussing the historical and cultural significance of Durga Puja in Bangladesh, each sentence, whether it's about the rituals or the economic impact, would be represented as a vector. These sentence vectors would then be processed by a custom transformer encoder, allowing us to capture the relationships between different aspects of the festival described in the article. This approach ensures that no crucial information is lost and that the complex semantic relationships within the long text are fully captured. The initial plan is to take a long text and slice it up into sentences based on the sentence-ending delimiter words and also extract each word from that sentence and pull out the semantics of the text based on the characteristics of their grammatical composite structure. This paper proposes a method that is a collaboration between Bert and a custom transformer encoder to build a model that will solve the problem that arises in calculating long text similarity. So as we said earlier we take the Bangla language as a research object so we use the Bangla-Bert-Base model which is a pre-trained language model that uses the Masked Language Model, Similar to another model like Bert, But pre-train it on the Bengali language and its nuances on the Bengali language specifically makes it special for our collaboration with that[11].

II. Methodology

Given two long bangla input text segments, we want to derive a similarity score at the semantic level. For this, our target is to make two semantically represented vectors for two input sequences which will contain more precise and accurate information packed with contextual information. Despite the previous methods, this more precise vector representation will help us to get a more precise and more accurate similarity score. From two long Bengali texts input processing to calculate semantic similarity score between In our proposed method, we break down our process into four parts- BERT-powered Text Preprocessing, Constructing the Custom Transformer Encoder, Generating Contextualized Vector Representations, and Computing Semantic Similarity. (Fig. 1)

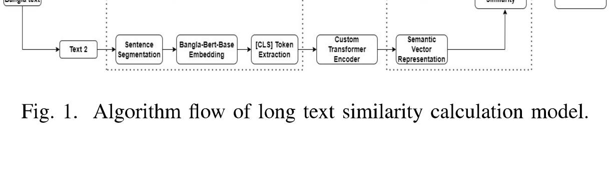


Fig. 1. Algorithm flow of long text similarity calculation model.

A. BERT-powered Text Pre-processing

In this process, the first work is to split the long input text into sentences. The sentence ending delimiter indicates the end of a sentence in the Bangla text. After splitting that makes a list of all sentences and stores it. Subsequently, loop through all the input sentences and convert the sentence into numerical representation using a tokenizer(Bangla-Bert-Base). Due to the use of Bert, we obtain dynamic word vectors rich in contextual information and sentence vectors containing richer information. After obtaining the contextualized word embeddings from the Bangla-Bert-Base, we extract the embedding corresponding to the [CLS] token, which serves as a summary representation of the entire sentence. Input sentences are passed through the tokenizer. There we process to make three vectors- word embeddings, position embeddings, and segmentation embeddings. In word embeddings, the tokenizer converts each word into a unique numerical ID by performing tokenization and creating a dictionary that contains token IDs for the sentence. In position embeddings, it captures sequential information. Position embeddings are typically created using sinusoidal functions that encode the position of each word as a vector. These vectors are then added to the word embeddings. In segmentation embeddings, token-type IDs are generated and a sequence that represents and distinguishes each sentence is created. Segmentation embeddings help the model understand the boundaries between sentences or segments, enabling it to process the input text more effectively and capture the relationships between different parts of the text. Since we are working on a long text it is an important part of the input stage and also this makes the end of the input preparation part.

B. Constructing the Custom Transformer Encoder

The encoder follows a typical Transformer encoder architecture, consisting of the following components in sequential order. Input Embedding, The encoder receives the input sequence as a combination of word, position, and segment embeddings, as detailed in the input preparation section. This embedding layer transforms discrete tokens into continuous vector representations, capturing semantic meaning, sequential order, and segment differentiation. Encoder Layers, this layer consists of multi-head attention, feed-forward network, layer normalization, and dropout. We replace the self-attention mechanism of the encoder with a multi-head attention mechanism this allows the model to attend to the different parts of the input sequence and capture the relationship between words. This mechanism helps the encoder to understand the context of each word's relationship between them with more implicit information. This mechanism allows the model to attend to different parts of the input sequence simultaneously and from multiple perspectives. Each attention head focuses on specific relationships between words or segments, capturing diverse aspects of the text's meaning. The outputs from multiple heads are then combined to form a richer representation. A feed-forward network, consisting of two fully connected layers with a ReLU activation function, further processes the output of the multi-head attention layer. This network extracts higher-level features and introduces non-linearity into the model, enabling it to capture complex relationships between words and segments. The core of the encoder is composed of multiple stacked encoder layers. Each layer employs a multi-head self-attention mechanism, a position-wise feed-forward network, layer normalization, and dropout to process the input embeddings and generate contextualized representations. These layers work together to capture relationships between words in the input sequence and enrich the representation of each word with contextual information. The encoder in our model is composed of 16 stacked encoder layers, each employing a multi-head self-attention mechanism with 16 attention heads, a position-wise feed-forward network with a size of 3072 (dff), layer normalization, and dropout to process the input embeddings and generate contextualized representations. The model's dimensionality is set to 768, determining the size of the embedding vectors. In addition to word and position embeddings, our encoder incorporates (Token Type Embeddings) to capture relationships between different sentences within the long text. These IDs help the model distinguish between sentences, enabling it to capture the complex semantic relationships within the long text more effectively. The final output of the encoder is a sequence of contextualized vector representations, where each vector represents a word in the input sequence, enriched with information from other words in the sequence. (fig 2)

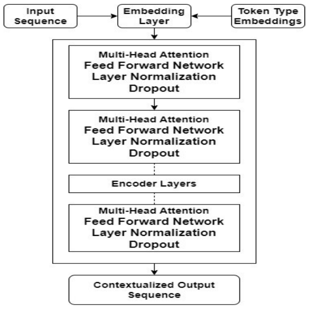


Fig. 2. Transformer encoder structure.

C. Generating Contextualized Vector Representations

The encoder adopts a standard transformer encoder architecture, comprising the following components in sequential order: Input Embedding, which receives the input sequence as a combination of word, position, and segment embeddings, transforming the discrete tokens into continuous vector representations that capture semantic meaning, sequential order, and segment differentiation; and Encoder Layers, which are composed of multiple stacked encoder layers that employ a multi-head self-attention mechanism, a position-wise feed-forward network, layer normalization, and dropout to process the input embeddings and generate contextualized representations. These layers work together to capture relationships between words in the input sequence and enrich the representation of each word with contextual information. The standard self-attention mechanism is replaced with a multi-head attention mechanism to enhance the model's ability to capture intricate relationships within the text. This mechanism allows the model to attend to different parts of the input sequence simultaneously and from multiple perspectives. Each attention head focuses on specific relationships between words or segments, capturing diverse aspects of the text's meaning. The outputs from multiple heads are then combined to form a richer representation.

$$\mathbf{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathbf{softmax}\left(\frac{\mathbf{Q}\cdot\mathbf{K}^{\mathbf{T}}}{\sqrt{\mathbf{d_k}}}\right)\cdot\mathbf{V}$$

where:

Q (Query), K (Key), and V (Value) are matrices representing different aspects of the input sequence.

d_k is the dimension of the key matrix, used for scaling.

A feed-forward network, consisting of two fully connected layers with a ReLU activation function, further processes the output of the multi-head attention layer. This network extracts higher-level features and introduces non-linearity into the model, enabling it to capture complex relationships between words and segments. Layer normalization is applied to stabilize training, and dropout is used to prevent overfitting, ensuring the model generalizes well to unseen data. After passing through all the encoder layers, the encoder produces an output tensor that contains the encoded representation of the input sequences. Each input sequence now has corresponding encoded vectors that capture its comprehensive meaning and context within the entire sequence.[fig 3]

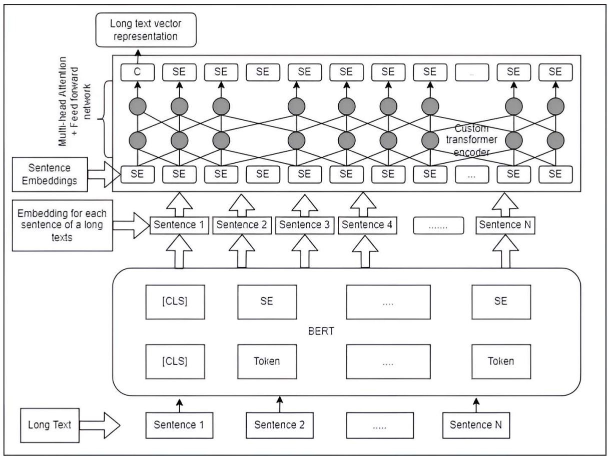

Long text vector representation
Multi-head Attention + Feed forward network
C
SE
...
Custom transformer encoder
Sentence Embeddings
Embedding for each sentence of a long texts
Sentence 1
Sentence 2
Sentence 3
Sentence 4
....
Sentence N
[CLS]
SE
....
BERT
Token
Long Text
.....

Fig. 3. Semantic vector representation process

D. Computing Semantic Similarity

As we obtained our text vector representation in the previous phase, that is the overall semantic representation of input texts. We can say that the similarity between these two vector representations is the semantic similarity between the input texts. For similarity calculation, we will use cosine similarity to get a similarity score.

$$\mathbf{Sim(A,B)} = \frac{\mathbf{A}\cdot\mathbf{B}}{||\mathbf{A}|| \times ||\mathbf{B}||} = \frac{\sum_{\mathbf{i=1}}^{\mathbf{n}}(\mathbf{A_i} \times \mathbf{B_i})}{\sqrt{\sum_{\mathbf{i=1}}^{\mathbf{n}}\mathbf{A_i^2}} \times \sqrt{\sum_{\mathbf{i=1}}^{\mathbf{n}}\mathbf{B_i^2}}}$$

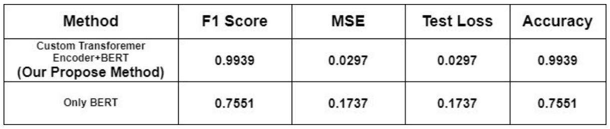
Where:

A and B are the vector representations of the texts.

Cosine similarity will find the angle between two vectors in the vector space as a measure of the difference between two individual texts and its value ranges from 0 to 1. The closer to 1 is more the two texts are similar and more closer to 0 is more the two texts are different.

III. Experiment and Results

In the experiment part, The proposed semantic similarity model was trained and evaluated on 5,000 Bangla text pairs from ”The Daily Ittefaq” news articles. Each pair consisted of an original article (Text 1) and a modified version (Text 2) created through paraphrasing, reordering, or adding/removing context. The pairs were annotated with similarity scores ranging from 0.0 (no similarity) to 1.0 (identical). The dataset was split into 80% training and 20% testing for model development. We take two standards to compare between them to clarify the better approach for calculating semantic similarity between two long Bangla texts. One approach is our proposed method where we propose an approach to calculating semantic similarity between two long Bangla texts by addressing the limitations of BERT, which is restricted to processing 512 tokens. The method involves extracting sentence embeddings from BERT and passing them through a custom transformer encoder designed to capture the full semantic context of longer texts. This encoder provides a more precise semantic vector representation that enhances the accuracy of similarity scoring. The other one is only Bert approach model performance was assessed using Test loss, Mean Squared Error(MSE), F1 Score, and accuracy metrics. Both method is tested on similar test data and the following table demonstrates the test results [Fig 4].



Method	F1 Score	MSE	Test Loss	Accuracy
Custom Transforemer Encoder+BERT (Our Propose Method)	0.9939	0.0297	0.0297	0.9939
Only BERT	0.7551	0.1737	0.1737	0.7551

Fig. 4. Evaluation matrices for two methods on common validation data.

The performance results demonstrate that the proposed method significantly outperforms the standard BERT approach in calculating semantic similarity. In our proposed approach, an F1 score of 0.9939 reflects an excellent balance between precision and recall. Achieved a test loss of 0.0297, indicating high precision in similarity scoring. On the other hand, the only BERT approach showed a significantly higher test loss of 0.1737 and a lower F1 score of 0.7551, underscoring its limitations in handling long texts effectively. In conclusion of experiment and results, our proposed method effectively addresses BERT's limitations regarding long texts, yielding superior performance in semantic similarity tasks for Bangla language texts.

Future work

While there has been research on short answer evaluation systems[12], no methods exist for long answer evaluation in Bangla, presenting an opportunity for automation in this area. Future research will also aim to expand the proposed method to support additional low-resource languages, adapting the custom transformer encoder and BERT model to each language's unique linguistic features and improving cross-linguistic semantic similarity assessments.

Conclusion

This paper proposes a novel method for calculating semantic similarity between long Bengali texts by integrating BERT with a custom transformer encoder. The approach uses sentence embeddings to build comprehensive semantic representations, addressing limitations in capturing context and relationships in lengthy texts. The custom encoder's multi-head attention and feedforward networks reduce information loss and input constraints, improving bidirectional semantic understanding. Experiments confirm its effectiveness, offering a robust solution for accurate semantic analysis in Bengali.

References

[1] Budan, I.; Graeme, H, “Evaluating wordnet-based measures of semantic distance,” Comutational Linguist. 2006, 32, 13–47.

[2] Sim, K.M.; Wong, P.T, “Toward agency and ontology for web-based information retrieval,” in IEEE Trans. Syst. Man Cybern. C Appl. Rev. 2004, 34, 257–269.

[3] Julian Risch, Timo Möller, Julian Gutsch, Malte Pietsch, “Semantic Answer Similarity for Evaluating Question Answering Models,” arXiv preprint arXiv:2108.06130, 2021 - arxiv.org.

[4] Dhivya Chandrasekaran and Vijay Mago, “On Evolution of Semantic Similarity – A Survey,” in ACM Computing Surveys 54(2):1-37 (2021)

[5] Md Shajalal and Masaki Aono, Semantic Textual Similarity in Bengali Text, in 2018 International Conference on Bangla Speech and Language Processing (ICBSLP)

[6] MD. Asif Iqbal, Omar Sharif, Mohammed Moshiul Hoque and Iqbal H. Sarker, “Word Embedding based Textual Semantic Similarity Measure in Bengali,” under CC BY-NC-ND license (http://creativecommons.org/licenses/by-n)

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” arXiv:1810.04805v2 [cs.CL] 24 May 2019.

[8] Xiao Li and Lanlan Hu, “Chinese long text similarity calculation of semantic progressive fusion based on Bert,” in Journal of Computational Methods in Sciences and Engineering, vol. 24, no. 4-5, pp. 2213-2225, 2024.

[9] Ming Ding, Chang Zhou, Hongxia Yang, Jie Tang, “CogLTX: Applying BERT to Long Texts,” Part of Advances in Neural Information Processing Systems 33 (NeurIPS 2020)

[10] F Incitti, F Urli, L Snidaro, “Beyond word embeddings: A survey,” in Information Fusion Volume 89, January 2023, Pages 418-436.

[11] M. Kowsher, Abdullah As Sami, Nusrat Jahan Prottasha, Mohammad Shamsul Arefin, Pranab Kumar Dhar, Takeshi Koshiba, “Bangla-BERT: Transformer-Based Efficient Model for Transfer Learning and Language Understanding,” Published in: IEEE Access (Volume: 10).

[12] Teddy Mantoro, Santhy David Nawi, Akeem Olowolayemo, and Y. Tagawa, “Short Answer Scoring in English Grammar using Text Similarity Measurement,” in 2018 4th International Conference on Computing, Engineering, and Design (ICCED)