



## ARTICLE

# Improving neural machine translation by integrating transliteration for low-resource English–Assamese language

Basab Nath<sup>1</sup> , Sunita Sarkar<sup>1</sup> , Somnath Mukhopadhyay<sup>1</sup> and Arindam Roy<sup>2</sup>

<sup>1</sup>Department of Computer Science and Engineering, Assam University, Silchar, India and <sup>2</sup>Department of Computer Science, Assam University, Silchar, India

**Corresponding author:** Sunita Sarkar; Email: [sarkarsunita2601@gmail.com](mailto:sarkarsunita2601@gmail.com)

(Received 13 December 2022; revised 5 January 2024; accepted 13 February 2024; first published online 27 May 2024)

Special Issue on ‘Natural Language Processing Applications for Low-Resource Languages’

## Abstract

In machine translation (MT), one of the challenging tasks is to translate the proper nouns and technical terms from the source language to the target language while preserving the phonetic equivalent of original term. Machine transliteration, an essential part of MT systems, plays a vital role in handling proper nouns and technical terms. In this paper, a hybrid attention-based encoder–decoder machine transliteration system is proposed for the low-resource English to the Assamese language. In this work, the proposed machine transliteration system is integrated with the previously published hybrid attention-based encoder–decoder neural MT model to improve the translation quality of English to the Assamese language. The proposed integrated MT system demonstrated good results across various performance metrics such as BLEU, sacreBLEU, METEOR, chrF, RIBES, and TER for English to Assamese translation. Additionally, human evaluation was also conducted to assess translation quality. The proposed integrated MT system was compared with two existing systems: the Bing translation service model and the Samanantar Indic translation model.

**Keywords:** low resource; neural machine translation; transliteration; named entities; gated recurrent units

## 1. Introduction

Communication plays a pivotal role in facilitating the exchange of knowledge and fostering social interactions among individuals. However, the multitude of languages spoken worldwide, exceeding 6500 in number, poses a formidable obstacle to seamless communication (Blasi, Anastasopoulos, and Neubig 2022). In the specific context of Assam, a region located in the north-eastern part of India, the predominant language is Assamese, which is extensively used throughout the expansive Brahmaputra Valley (Baruah *et al.* 2014). Tracing its origins back to the seventh century, Assamese language has experienced significant advancements as a written system, which is a branch of the Indo-Aryan language. To enable effective communication between several regional languages, the automated translation system becomes imperative. Machine translation (MT) system has provided a way to lower the language barrier in commutation (Al-Muzaini, Al-Yahya, and benhidour 2018). However, achieving a high level of accuracy in translation requires an extensive vocabulary encompassing both the source and target languages, a comprehensive understanding of the semantic properties inherent in both languages, and other relevant considerations. Viable models for MT include rule-based, statistical-based, and neural network-based approaches

(Cho *et al.* 2014). Rule-based methods rely on maintaining vast dictionaries containing predefined semantic and syntactic rules of the languages concerned. Statistical models employ probabilistic functions to construct tables with semantic rules learned from bilingual corpora. These statistical tables may include syntax trees or translation models for phrases. With the advent of neural machine translation (NMT) models, tasks such as developing parse trees, phrase translation models, rule-based dictionaries, and other feature engineering activities have become significantly easier.

NMT aims to construct and train a single, expansive neural network capable of reading a sentence and producing an accurate translation (Luong, Pham, and Manning 2015). MT refers to the automated process of using computers to translate text from one language to another. While translating from one language to another, handling named entities such as names of people, places, medicines, and sports terms is a challenging task as they are the same in all languages and conserve their phonetics.

According to Karimi, Scholer, and Turpin (2011), “transliteration” is the process of transferring any word from one language to another without changing the phonetics characteristics of the original term. For example, while transliterate from English to Assamese the word “moon” becomes “মুন”(moon), “lovely” becomes “লাভলী” (lovely), “kingfisher” becomes “কিংফিছাৰ” (kingfisher) and for many more named entities. The primary objective of this research is to propose a methodology aimed at enhancing the quality of translations from English to Assamese by incorporating transliteration techniques for named entities. In this paper, a GRU-based transliteration system for English to Assamese is developed and integrated with translation system for English to Assamese.

This paper has following contributions:

- Building a model based on encoder–decoder architecture to achieve transliteration. The proposed model uses a hybrid attention mechanism for an appropriate alignment of input–output tokens. Byte-pair encoding (BPE) is also used to tokenize the vocabulary into subword tokens. The experimental results show that the frequency of the out-of-vocabulary (OOV) issue is reduced because of subword tokenization.
- Developing an integrated system combining translation and transliteration model to achieve high-quality translations for English to Assamese.
- Evaluation of MTmodel using several metrics.

This paper is structured as follows: In Section 2, we provide a comprehensive literature review on machine translation and transliteration. In Sections 3 and 4, we elaborate on the background research and methodology employed in this study. In Section 5, we present our experimental setup and results and compare our approach with existing systems. Finally, in Section 6, we present a summary of our findings and offer concluding remarks.

## 2. Literature review

In recent years, there has been a lot of research and development in the fields of machine translation and transliteration. The encoder–decoder architecture, which serves as the foundation for most NMT algorithms, has demonstrated promising results in various language pairings (Baruah *et al.* 2014). This approach has gained acceptance and implementation in renowned organizations such as Baidu, (Zhou *et al.* 2016), Google (Yonghui *et al.* 2016), and Microsoft (MSFT 2020).

Numerous papers have made valuable contributions to the improvement of NMT. For instance, a notable contribution by Cho *et al.* (2014) introduced an RNN encoder–decoder model that effectively converts sequences of varying lengths into fixed-length vectors and decodes them to generate desired phrases. This technique provides valuable insights into sequence conversion and

fixed-length vector representation. Addressing the vocabulary problem, Sennrich *et al.* (2016) proposed grouping uncommon words into subword units, offering a solution for handling vocabulary variation. Furthermore, He *et al.* (2016) integrated statistical machine translation (SMT) features into the log-linear framework, opening opportunities for further advancements in NMT. Another noteworthy approach by Cheng (2019) explored semi-supervised learning using auto-encoders and sequence-to-sequence models, enabling the creation of multi-way NMT models. In addition to these contributions, Laskar *et al.* (2022) focused specifically on improving NMT for the low-resource English–Assamese language pair. Their research aimed to enhance translation quality by proposing novel techniques tailored to the challenges posed by limited resources.

Decoding methods also play a crucial role in enhancing translation quality. Studies such as Yuchen *et al.* (2018) have introduced techniques like model assembling, averaging, and candidate re-ranking in order to enhance the performance of NMT models. Additionally, Shao and Nivre (2016) demonstrated the effectiveness of convolutional neural networks (CNNs) in English-to-Chinese transliteration, surpassing traditional PBSMT methods in terms of accuracy. Moreover, Jankowski *et al.* (2021) presented a paper focusing on multilingual NMT models tailored for Slavic languages, including Polish and Slovak. The paper introduced soft decoding, a technique that allows the NMT model to generate multiple translations simultaneously. These research endeavors have significantly contributed to the advancements and achievements in machine translation and transliteration, paving the way for improved translation quality and expanded capabilities in various language pairs.

Advancements in the field of transliteration have also been substantial, with researchers making significant contributions in this area. Notably, Bahdanau and Bengio (2014) proposed bidirectional encoder and attention-based decoder models for transliteration tasks across different language pairings, highlighting their effectiveness and advantages over conventional phrase-based statistical machine translation (PBSMT) methods. In the context of low-resource languages, Le *et al.* (2019) introduced multiple neural network-based techniques to address the challenges of transliteration, offering valuable guidance for handling transliteration tasks in scenarios with limited resources. Similarly, Hadj Ameer *et al.* (2017) developed an attention-based, bidirectional encoding and decoding model specifically designed for Arabic-to-English machine transliteration, demonstrating its competitiveness with state-of-the-art sequence-to-sequence models. The research conducted by Grundkiewicz and Heafield (2018) focused on NMT techniques for named entity transliteration, presenting and exploring various approaches to enhance transliteration quality, including leveraging NMT models with attention mechanisms. Other studies, such as Younes *et al.* (2018) and Masmoudi *et al.* (2020) have also investigated effective transliteration techniques for diverse language pairs, encompassing the Romanization of the Tunisian language. Additionally, Makarov *et al.* (2020) addressed the challenges associated with low-resource transliteration for Ukrainian, conducting an investigation into unsupervised methods for transliteration mining and providing valuable insights into the utilization of data-driven approaches. Furthermore, the paper by Džambazov *et al.* (2019) delved into the realm of neural transliteration for Slavic languages, specifically Ukrainian.

Although the studies mentioned above primarily focus on neural machine translation and transliteration, it is worth noting several other relevant research endeavors. For instance, Lakshmi and Shambhavi (2019) employed a character-level bidirectional long short-term memory (BiLSTM) model to enhance English–Kannada transliteration accuracy. In a similar vein, Abbas and Asif (2020) developed a hybrid technique for Hindi-to-English transliteration, achieving an impressive transliteration accuracy of 97%. Furthermore, Vathsala and Holi (2020) utilized recurrent neural networks (RNNs), specifically long short-term memory (LSTM) models, to detect inappropriate content on social media by means of transliterating and translating Twitter data, thereby demonstrating the superiority of RNN-LSTM models over SMT models in this context. Additionally, Athavale *et al.* (2016) an LSTM-based technique was proposed for named entity recognition (NER) without language-specific restrictions, surpassing the performance of

rule-based systems. Moreover, Kaur and Singh (2015) introduced an algorithm for translating handwritten text into the International Phonetic Alphabet (IPA), effectively shedding light on the limitations of grapheme-based transliteration. Notably, Kaur and Goyal (2018) presented a remarkably accurate Punjabi-to-English transliteration method, employing SMT techniques that achieved an impressive accuracy rate of 96% across various source–target language combinations. Lastly, Hany Hassan *et al.* (2018) discussed the achievements of Microsoft’s MT system, demonstrating its parity with professional human translations for Chinese–English language pairs and its superior performance in comparison to crowdsourced references.

Collectively, these studies aid in the advancement of neural machine translation and transliteration techniques, addressing various challenges and providing valuable insights into enhancing translation and transliteration accuracy across different language pairs and resource scenarios. However, to our knowledge, there have been limited attempts to integrate transliteration and translation into a single system for the low-resource English–Assamese language pair. In the subsequent section, we will provide an overview of the requisite background information, encompassing the translation–transliteration model and the integration framework.

### 3. Preliminaries

In this section, we have discussed encoder–decoder models such as LSTM, GRU, and transformer, as well as the significance of BPE in addressing challenges like OOV issue and inflection. Additionally, we have also presented a comparative analysis of these models, highlighting their key features.

#### 3.1 Encoder–decoder models

The encoder–decoder (Bahdanau *et al.*, 2014; Laitonjam and Singh 2022) models work in a sequence-to-sequence manner. The input sequence is compressed into a context vector by the encoder, and the decoder utilizes this context vector to reconstruct the target sequence.

In the encoder, the hidden state  $h_t$  is computed using the current input vector  $x_t$  and the previous encoder hidden state  $h_{t-1}$ . This is represented as  $h_t = E_A(h_{t-1}, x_t)$  (Cho *et al.* 2014), where  $E_A$  is an activation function.

In the decoder, the hidden state  $s_t$  is computed using the previous decoder hidden state  $s_{t-1}$ , the previous decoder output  $y_{t-1}$ , and the context vector  $c$ . It is represented as  $s_t = E_A(s_{t-1}, y_{t-1}, c)$  (Phan-Vu *et al.* 2019), where  $E_A$  is an activation function.

Understanding the specific architectures like LSTM, gated recurrent unit (GRU), and transformer necessitates a thorough grasp of the explanations of the encoder and decoder components.

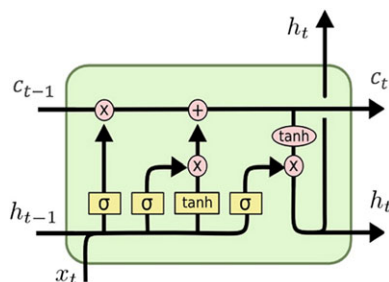
##### 3.1.1 Long short-term memory (LSTM)

The LSTM network, as detailed by Hochreiter and Schmidhuber (1997), represents an evolution of the conventional RNN design. It overcomes the limiting factor of vanishing gradients inherent in standard RNNs, thereby enhancing the model’s ability to learn and sustain long-term dependencies. The LSTM cell has several gates that control the flow of information, including the input gate ( $i$ ), forget gate ( $f$ ), and output gate ( $o$ ). The formula for the LSTM cell’s hidden state ( $h_t$ ) is determined as follows:

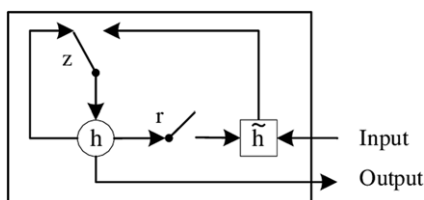
$$h_t = o \cdot \tanh(c_t)$$

where

- $h_t$  represents the hidden state at time step  $t$ .
- $o$  is the output gate’s activation, which controls the amount of information to be passed to the output.



**Figure 1.** Long short-term memory(LSTM) architecture (Hochreiter and Schmidhuber, 1997).



**Figure 2.** Gated recurrent unit (GRU) architecture (Cho et al. 2014).

-  $c_t$  is the cell state at time step  $t$ , computed using the input gate ( $i$ ), forget gate ( $f$ ), and a combination of the current input ( $x_t$ ) and the previous hidden state ( $h_{t-1}$ ).

The LSTM architecture incorporates a gating mechanism that empowers the model to selectively retain or discard information. This capability enables the LSTM to effectively handle long sequences and maintain contextual dependencies during the translation process. Figure 1 illustrates the structure of the LSTM model.

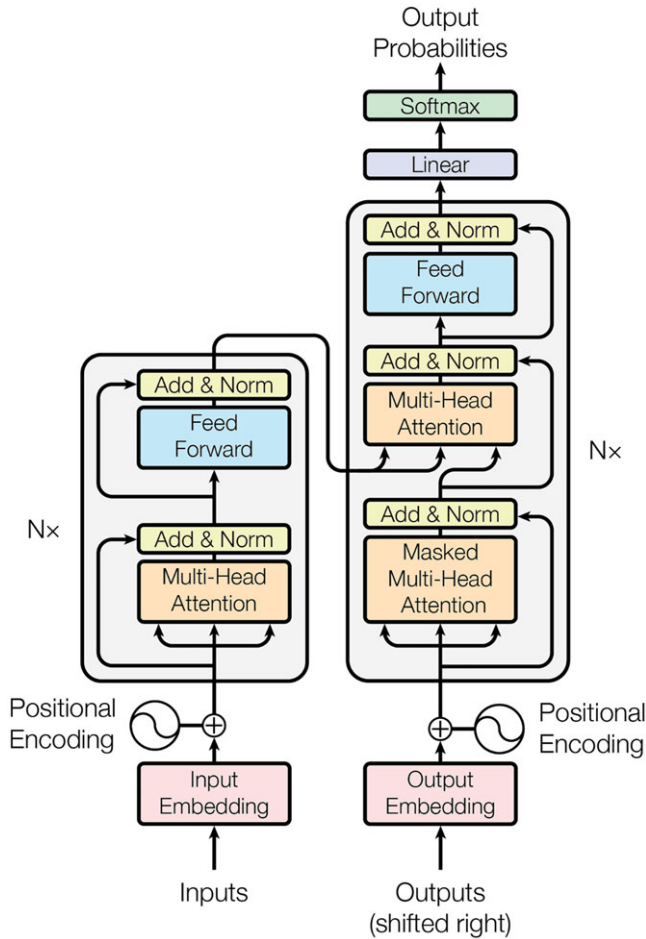
### 3.1.2 Gated recurrent units (GRUs)

GRUs are a type of RNN architecture that were introduced in 2014 by Cho *et al.* (2014) as a simpler alternative to LSTM units. GRUs consist of two main components, namely the update and reset gates. The purpose of the update gate is to regulate the quantity of prior information that is to be retained, whereas the reset gate's role is to dictate the extent of forgetting or ignoring the previous hidden state.

The formula for a GRU unit can be written as follows:

- Reset gate:  $r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$
- Update gate:  $z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$
- Candidate activation:  $\tilde{h}_t = \tanh(W x_t + r_t \circ (U h_{t-1} - 1) + b)$
- Hidden state:  $h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \tilde{h}_t$

Here,  $x_t$  is the input at time step  $t$ ,  $h_{t-1}$  is the hidden state at the previous time step,  $r_t$  and  $z_t$  are the reset and update gate activations, respectively, and  $\tilde{h}_t$  is the candidate activation.  $\circ$  signifies the operation of element-wise multiplication, while  $\sigma$  represents the sigmoid activation function and  $\tanh$  corresponds to the hyperbolic tangent activation function.  $W$ ,  $U$ , and  $b$  are the learnable weights and biases of the network. The architecture of GRU is depicted in Figure 2.



**Figure 3.** Transformer architecture (Vaswani *et al.* 2017).

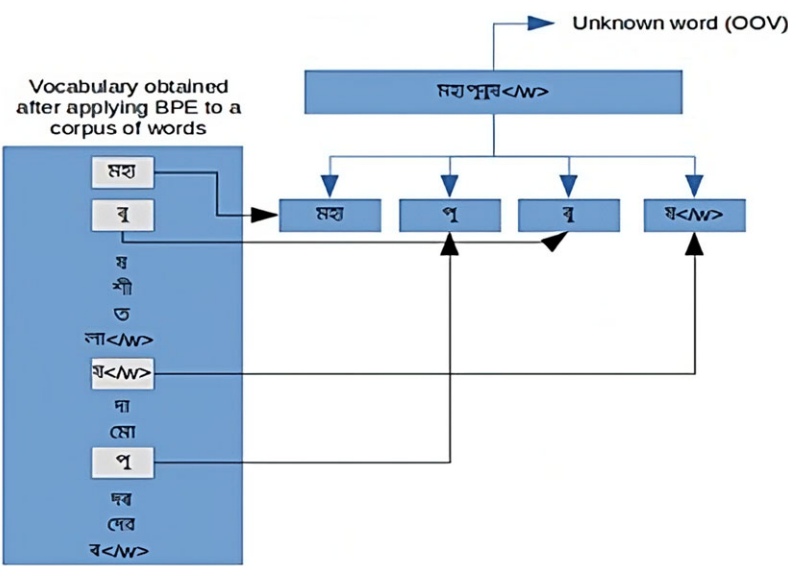
### 3.1.3 Transformer

The architectural design known as the transformer was first proposed by Vaswani *et al.* (2017), revolutionized sequence-to-sequence tasks by replacing RNNs with self-attention mechanisms. The transformer architecture is comprised of two major parts: an encoder and a decoder, both of which are constructed using multiple layers. In contrast to conventional recurrent neural networks, transformers process the input sequence in parallel, making them highly efficient for long sequences. In the encoder, self-attention is used to compute attention scores between query (Q), key (K), and value (V) embeddings of each word. These scores are then used to weight the value embeddings, producing the encoded representation of the input sequence.

In the decoder, in addition to self-attention, the transformer that employs encoder-decoder attention emphasizes on various elements of the input's encoding sequence while generating the output sequence. The transformer also incorporates positional encodings to account for the sequential order of the words. Figure 3 illustrates the transformer architecture. Table 1 summarizes the technical differences between LSTM, GRU, and transformer models for sequence-to-sequence tasks (Bahdanau *et al.*, 2014; Cho *et al.*, 2014; Vaswani *et al.*, 2017).

**Table 1.** Comparison of LSTM, GRU, and transformer

Model	Architecture	Features
LSTM	Recurrent neural network (RNN)	Input, forget, output gates
GRU	Recurrent neural network (RNN)	Update, reset gates
Transformer	Non-recurrent	Self-attention



**Figure 4.** Working of BPE solving OOV issue.

**3.2 Byte-pair encoding (BPE)**

In information theory, BPE is a compression technique that iteratively replaces the most frequent consecutive byte pairs (strings) in the data with codes that are not present in the original data. In NLP, BPE is used as a subword tokenization technique. A variant of this approach finds its application in models for natural language interpretation, such as LSTM, GRU, transformers, and GPT-2 for tokenizing word sequences. BPE is used during the preprocessing of data. In BPE, a word is segmented into individual characters. BPE maintains a counter for all the unique words in the vocabulary. For example, there is a vocabulary of words such as smart, smarter, oldest, and wildest, with frequencies of 3, 4, 6, and 3, respectively. In BPE, these words are split into characters, and the vocabulary is formed along with their frequency (Ramesh *et al.* 2022). The whole process is shown in Figure 4 for a better understanding:

$$smart < /w > : 3, smarter < /w > : 4, oldest < /w > : 6, wildest < /w > : 3$$

A fixed number of iterations is run, and at every iteration, the frequency of each consecutive byte pair is counted. The pair with the highest frequency is combined as a single token. In this example, it is observed that the words “oldest” and “wildest” have occurred six and three times, respectively. After all the words are tokenized as individual characters, the first iteration of the process of generating byte-pair tokens is started. When the iteration begins, consecutive pairs of characters are checked to compute the pair that has the highest frequency. It is seen that the consecutive pair of characters “e” and “s” is the pair with the highest occurrence (6 + 3 = 9), which comes from the words “oldest” and “wildest”. Hence, the characters “e” and “s” are combined as a



single token, “es”. The process of combining the most frequent pairs continues until a maximum number of iterations is reached.

One of the advantages of BPE is that it solves OOV problems to an extent because the vocabulary prepared by BPE consists of subword tokens instead of whole words. Unknown words are split using the subword tokens in the vocabulary, hence eliminating the OOV issue.

#### 4. Proposed methodology

The proposed model integrates a translation model with the transliteration model for low-resource English–Assamese languages using a hybrid attention mechanism. Accordingly, the proposed methodology has the following four steps:

1. **Preprocessing** : We preprocess the corpus by normalizing the text to all lowercase letters, removing special characters, and so on.
2. **Developing a translation model**: We develop a MT model based on GRU with a hybrid attention mechanism.
3. **Developing a transliteration model**: We develop a GRU-based machine transliteration model with a hybrid attention mechanism.
4. **Integrating the translation model with the transliteration model**: Finally, we integrate the above two models to build a complete translation system.

##### 4.1 Translation model

The NMT model utilized in this research adopts an encoder–decoder architecture, incorporating a hybrid attention mechanism (Nath, Sarkar, and Das 2022). The encoder and decoder components consist of unidirectional, single-layered GRUs with 1024 hidden units within each layer. A custom bilingual corpus of sentences from the target and source languages is used for training the model. Additionally, the decoder is composed of 1024 units in the final output layer.

This research uses a hybrid attention mechanism and an encoder–decoder architecture to create both translation and transliteration model (Nath *et al.* 2022). Hybrid attention selects the attention mechanism among additive and multiplicative processes for a Bilingual corpus. During the initial learning phase of the NMT model, both additive and multiplicative attention processes are used to provide prediction output. Hybrid attention mechanism’s main idea is to find out the average loss produced by additive and multiplicative attention mechanisms individually. Then the minimum loss obtained from the average losses of additive and multiplicative attention mechanisms is used to update the weights of the model’s network. The average loss is calculated by taking the mean (average) of the losses obtained from two different attention mechanisms.

##### 4.2 Model network for transliteration

The transliteration model Shao and Nivre (2016); Younes *et al.* (2020) comprises a unidirectional single-layer encoder and decoder with GRUs with 512 hidden units per layer. Figure 5 illustrates the encoder–decoder architecture employed in the transliteration model. The model is trained on a bilingual corpus. The bilingual corpus of words is first converted to parallel subword tokens. The vocabulary is prepared from the corpus, and the embedding vectors are calculated accordingly. The Softmax layer in the decoder consists of 512 units with a sparse categorical cross-entropy loss function. BPE is used to attain subword tokens from the words of the corpus. In a machine transliteration model, to learn the relationship between parallel words, individual character-to-character mapping is learned. Hence, the atomic units considered here are characters in the words. Similarly, the atomic units of mapping are taken as subword tokens to get a more accurate translation for named entity words. The details of subword tokens are given in Table 2.



Table 2. Subword tokens

Pu	পু
ja	জা
sh	শী
it	ই
ala	লা

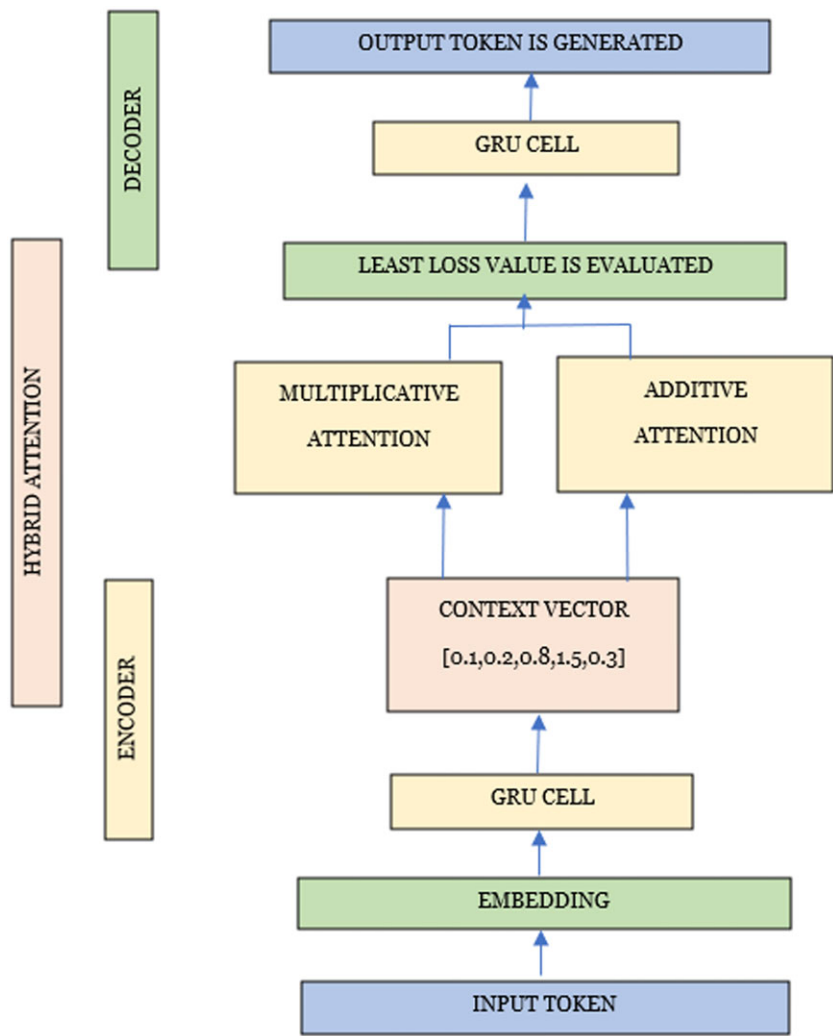


Figure 5. Encoder–decoder architecture for transliteration model.

- The steps for the transliteration model are illustrated with an example and are given below:
1. An input word is taken, such as “Parker.”
  2. The word is split into characters or pairs based on corresponding phonetic representation, such as “P a r k e r”.

3. The characters are then tokenized using BPE.
4. The input “Parker” is fed into the encoder component and converts it into a vector [0.3, 0.1, 0.2, . . . .].
5. The decoder receives the numerical representation and converts it into the corresponding word in the target language. “পাৰ্কাৰ”.
6. The output is then displayed, alike “পাৰ্কাৰ”.

### 4.3 Translation model integrated with transliteration

The integrated MT model consists of both a translation and a transliteration model. The model is trained independently in two distinct phases. While the training of the models was done independently in two phases, the models shared the same session during the training process. This allows seamless integration of the transliteration model into the inference pipeline of the NMT model. During evaluation, the transliteration model is invoked when named entities are detected in the input text. We employ subword tokenization in the transliteration model to handle OOV words like proper nouns that are not explicitly part of the model’s training vocabulary. This enables the transliteration of unseen words. The translation model executes the task of translating, while the transliteration model is specifically tasked with transliterating named entities. Figure 6 depicts the proposed integrated translation–transliteration system architecture. A nonlinear alignment between the positions of source words and corresponding target words poses a significant challenge for the precise placement of translated named entities within the target sequence.

In order to solve this problem, we used the attention weights provided by the attention layer in the translation model. This layer is used by the decoder to attain the context vector, which is essential for predicting the output sequence. The attention mechanism relies on encoder output and encoder hidden states for its computation of attention weights, which in turn facilitate the formation of the context vector.

The attention weights act as indicators of the relevance of a particular word within the source sequence during the process of translation. As a result, the utilization of attention weights allows the decoder to focus on specific words at specific instances of time, leading to a more precise translation.

We leveraged the attention layer’s functionality within the translation model via the utilization of its attention weights or attention scores. These scores are computed using the previous decoder hidden state ( $h_t$ ) and each encoder hidden state ( $h_s$ ) and this is represented as:

$$\alpha_{ts} = \frac{\exp(\text{score}(h_t, \bar{h}_s))}{\sum_{s'} \exp(\text{score}(h_t, \bar{h}_{s'}))}$$

These alignment scores indicate how much attention or emphasis should be placed on each word in the source sequence during translation.

The attention mechanism then combines these alignment scores to create an alignment vector, which has the same length as the source sequence. Each value in this vector corresponds to the importance (or probability) of the corresponding word in the source sequence. This helps the decoder decide what to focus on at each time step. The attention weights help identify the most focused input token by finding the index of the maximum value in the attention weight vector. This index is used to retrieve the corresponding word from a pre-built data structure mapping integers to words.

We employ the NER tool from SpaCy to determine whether the identified word is a named entity. If the word is classified as a LOCATION or ORGANIZATION type entity by the NER model, we check if a standard Assamese translation exists in our custom databases of common country names and institution names (e.g. “Bharot” for India). If a translation is available, we use that standard translation in the output sequence.

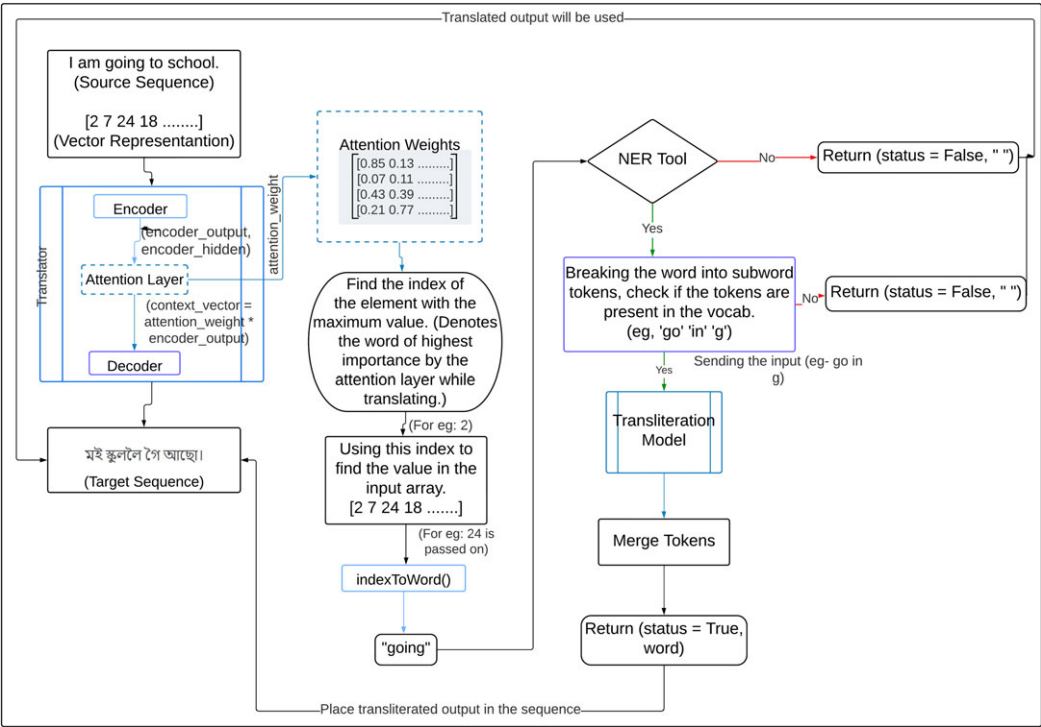


Figure 6. Proposed integrated translation-transliteration system architecture.

However, not all LOCATION and ORGANIZATION entities have an existing standard Assamese translation. For named entities that do not have a standard translation available in our databases, as well as other types of named entities like PERSON, we send the word through our transliteration model to produce an appropriate Assamese transliteration in the output sequence. For words not identified as named entities by the NER tool, we directly use the translated output from our translation model in the output sequence.

The algorithm is described in detail below. The following steps illustrate how an input sentence is translated by the integrated MT model.:

At each time instant  $t = i$ , where  $i$  ranges from 0 to the size of the source sentence:

1. Padding is applied to the input sentence to align its length with the expected source sentence length. Subsequently, the input tokens within the source sentence are converted into their corresponding unique integers, as defined during the vocabulary preparation phase.
2. The input sequence is then passed to the encoder component of the translation model. The encoder produces the encoder output and the encoder hidden state.
3. The decoder component of the translation model incorporates an attention layer, which takes as input the encoder output and encoder hidden state. The attention layer with the help of its fine-tuned attention weights computes the context vector. The context vector is obtained by multiplying vectors of the same shape, namely the encoder output and the attention weight.
4. The decoder receives the context vector and attention weights from the attention layer.
5. As the attention weight signifies the relative importance of each source word at a particular instant, we determine the index of the maximum value from the attention weight vector. The attention weights are represented as:

$$\begin{bmatrix} 0.12 & 0.0085 & \dots & 0.61 \\ 0.7 & 0.003 & \dots & 0.01 \\ \vdots & \vdots & \ddots & \vdots \end{bmatrix}$$

where each row represents a source token, and the number of columns corresponds to the encoder hidden state.

6. This index is utilized to retrieve the corresponding source token from the input array. This particular source token represents the focal point of translation at instant  $t = 0$ .
7. Subsequently, we pass on this source token to the NER checker tool. In the case where the input token is identified as a named entity, we further break it down into subwords. If these subwords exist within the vocabulary, the subword sequence is then forwarded to the transliteration model. The output subwords generated by the transliteration model are merged to form a single word. This resulting word is placed in the output sequence at the corresponding position for instant  $t$ .
8. If the input token is not identified as a named entity, or if the subwords are absent from the vocabulary, we employ the decoder component of the translation model to compute the translated output using the context vector. The translated output token is subsequently positioned within the output sequence at the respective position for instant  $t$ .
9. The maximum length of each iteration of these steps corresponds to the maximum length of the target sequence. Also when the “end of sentence” token ( $\langle \text{eos} \rangle$ ) is found out, the prediction ends.

## 5. Result and analysis

### 5.1 Data preprocessing

The corpus used to train both the translation and the transliteration models is taken from TDIL (TDIL-DC 2006), and in addition to that, we have increased data by 11% using back translation as well as back transliteration where required. We manually generated our back transliteration and translation data using the Aksharantar (ai4bharat 2022) and Bing websites (MSFT 2020). Over 7% of our transliteration and 4% of our translation corpora were generated by back-transliterating and translating a selection of English words and sentences that weren't included in the original corpus. The integrated model uses a corpus of parallel sentences which belong to various domains such as agriculture, entertainment, and history. All unwanted and irrelevant characters have been removed from the bilingual corpora. Both the corpora words have been converted to lowercase. Ninety-five percent of the corpus is used for training, while 5% is allocated for testing. The validation set is of the same size as the test set and is derived from an external dataset (ai4bharat 2022). All the words (from both the language pair) in the corpus are converted to a sequence of subword tokens. The bilingual corpus for the integrated translation–transliteration model undergoes padding for shorter sentences to make all the sentences of a particular language pair equal. This is done so that the model may be trained in batches. All the punctuations have been retained in this corpus. If there is no white space between a word and punctuation, then white is given in between them to consider the punctuation as an individual character. The corpus statistics for the translation and Tables 3 and 4 present the transliteration models.

We utilized 1200 merge operations in our models. This value was selected based on testing a range from 500 to 2000 merge ops. We used two metrics to evaluate the performance of the algorithm: accuracy and runtime. We found that the accuracy of the algorithm was maximized at 1200 merge operations. The runtime of the algorithm also decreased as the number of merge operations increased, but only up to a point. After 1200 merge operations, the runtime of the

**Table 3.** Corpus statistics for translation model

TDIL corpus	73,294
Manually generated	8,253
Total number of parallel sentences	81,547
Vocabulary count for English language	40,343
Vocabulary count for Assamese language	60,732
Total number of parallel sentences in the training set	77,470
Total number of parallel sentences in the testing set	4,077
Total number of parallel sentences in the validation set	4,077
Minimum word count in a sentence (both English and Assamese)	1
Maximum word count in a sentence (both English and Assamese)	22
Number of merge operations	1200

**Table 4.** Corpus statistics for transliteration model

TDIL corpus	61,960
Manually generated	6,945
Total number of parallel words	68,905
Vocabulary count for English language (subword tokens)	2027
Vocabulary count for Assamese language (subword tokens)	2070
Total number of parallel words in the training set	65,461
Total number of parallel words in the testing set	3,444
Total number of parallel words in the validation set	3,444
Number of merge operations	1200

algorithm started to increase again. Based on these results, we determined that 1200 is the optimal number of merge operations for our specific dataset and algorithm.

**5.2 Model training**

The proposed model is a GRU-based translation and transliteration model, trained on a system with the following hardware configuration: 24 GB of RAM and single-core, hyper-threaded Xeon processors with a clock rate of 2.3 GHz. Additionally, the system features a Tesla K80 GPU instance with 2496 CUDA cores and 12 GB of VRAM.

We have also trained an LSTM-based translation and transliteration model with the same parameters as the GRU model, as detailed in Table 5. In addition to the GRU and LSTM models, further experiments were conducted by training a transformer model. The transformer model shares parameters with both the GRU and LSTM models, keeping consistency in our methodology. The detailed parameters for the transformer-based model are provided in Table 6. By

**Table 5.** Parameters for LSTM and GRU-based translation and transliteration models

Parameters	Translation model (LSTM/GRU)	Transliteration model (LSTM/GRU)
Type of recurrent neural network instance	LSTM/GRU	LSTM/GRU
Encoder/decoder layers	1	1
Hidden units per layer	1024	512
Size of batch	32	16
Dimension of word embedding	300	300
Epochs count	16	15
Steps per epoch, number	1815	4092
Adoption of an optimizer	Adam	Adam
Initial rate of learning	0.001	0.001
Dropout factor	0.5	0.0
The function of loss	Crossentropy of categorical sparse	Crossentropy of categorical sparse

**Table 6.** Parameters for transformer-based translation model

Parameters	Transformer-based translation model
Type of recurrent neural network instance	Transformer
Encoder/decoder layers	1
Size of batch	32
Number of attention heads	2
Epochs count	15
Feed-forward dimension	256
Initial rate of learning	0.01
Dropout factor	0.5
Embedding dimension	300

adopting this methodology, we aim to facilitate a better comprehension of the model's behavior and interpretation of the results.

### 5.3 Evaluation metrics

The evaluation metrics used are BLEU (bilingual evaluation understudy) (Papineni *et al.* 2002), SacreBLEU (Post, Vilar, and Rebele 2018), TER (translation edit rate) (Snover *et al.* 2006), METEOR (metric for evaluating translation with explicit ordering) (Lavie and Denkowski 2009), chrF (character F-score) (Popović, 2015), and RIBES (reference independent BLEU estimation score) (Snover *et al.* 2006). In addition to these automated metrics, human evaluation was also conducted to assess translation quality. For TER, lower scores in these evaluation metrics indicate higher prediction accuracy.

Table 7. Performance of transliteration model

Model	ACC
GRU-based transliteration model	0.523

Table 8. Performance of distinct models for translation English to Assamese language

Model	BLEU score	sacreBLEU	METEOR	TER	chrF2	RIBES
Transformer model	22.00	24.00	29.00	75.00	36.00	44.00
GRU-based translation model	26.10	29.72	34.40	72.30	39.17	47.17
LSTM-based translation model	24.12	26.60	31.03	73.50	37.31	45.21
LSTM-based translation model with hybrid attention	25.60	28.30	33.60	72.70	38.30	46.60
LSTM-based translation model integrated with transliteration (Bahdanau attention)	27.30	30.00	35.10	69.70	41.00	49.00
LSTM-based translation model integrated with transliteration (hybrid attention)	27.80	31.70	35.40	69.00	40.40	48.70
GRU-based translation model along with hybrid attention	26.90	30.00	34.97	73.58	39.78	47.91
GRU-based translation model integrated with transliteration (Bahdanau attention)	26.79	29.82	34.22	73.17	39.88	47.78
<b>GRU-based translation model integrated with transliteration (hybrid attention)</b>	<b>28.13</b>	<b>32.22</b>	<b>35.76</b>	<b>69.44</b>	<b>40.76</b>	<b>49.23</b>

5.4 Experimental results

In this study, the transliteration and translation models were developed and evaluated independently. Subsequently, the translation model was integrated with the transliteration model, and the combined system was tested on 4,077 sentences. In Table 7, the transliteration model’s performance is assessed using the accuracy metric introduced by Zhang *et al.* (2016), denoted as ACC (accuracy). ACC measures the correctness of the transliteration output, providing insights into the model’s ability to accurately capture transliteration nuances based on the criteria established by Zhang et al. The experimental results for the standalone transformer model, GRU-based translation model, GRU-based translation model with a hybrid attention mechanism, proposed integrated system along with the Bahdanau attention mechanism, hybrid attention mechanism, as well as the LSTM-based translation models are presented in Table 8. The graphical representation in Figure 7 illustrates the progress of our model’s training through the epoch vs loss graph. The presented visual representation illustrates the gradual decrease in loss metrics over multiple epochs, indicating the model’s iterative improvement in error minimization and optimal fitting of the training dataset. The decision to utilize GRU in our translation model is driven by its competitive performance and potential ease of training.

The integration of a transliteration model with a GRU-based translation model, along with a hybrid attention mechanism, results in improved performance compared to the base translation model, as evidenced by various evaluation metrics. Specifically, the integration of the transliteration model leads to overall improvements in translation quality, particularly for named entities, resulting in improved scores across various evaluation metrics. The performance comparison graph is depicted in Figure 8, while Tables 9, 10, and 11 present output samples from a standalone transliteration model and translation samples from the integrated system, respectively.



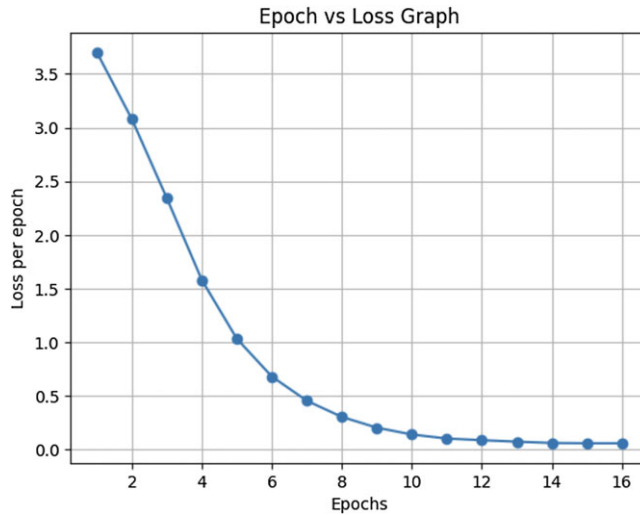


Figure 7. Epoch vs loss graph – training progress and error minimization.

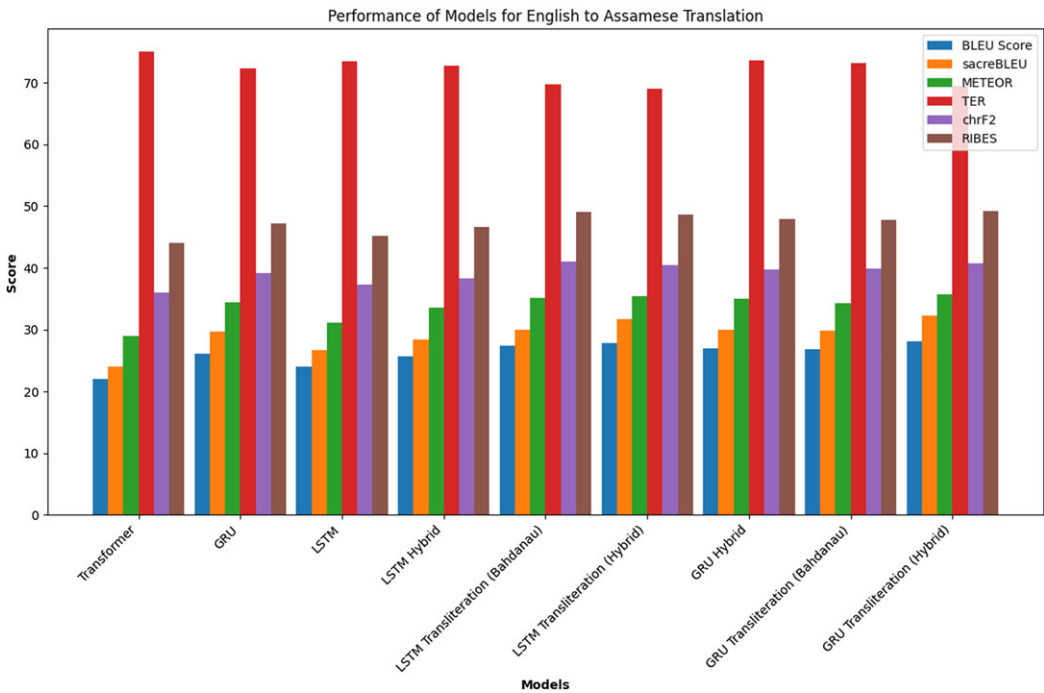


Figure 8. Bar chart for performance comparison I.

In Table 9, the \*-marked words are OOV, and the transliteration model recognizes and transliterates them. Hence, the OOV issue is solved by the transliteration model with the help of subword tokenization.

In this study, we compare the performance of our proposed integrated system with two existing MT approaches: Samanantar (Ramesh et al. 2022), which is based on transformer-based models trained using Fairseq, and the commercial Microsoft MT system (MSFT 2020). The Samanantar

Table 9. Sample outputs from standalone transliteration model

Input	Actual output	Predicted output
Usha	ঊষা (Usha)	আশা (Asha)
Kailashnath	কৈলাশনাথ (Kailashnath)	কৈলাশনাথ (Kailashnath)
Islam	ইছলাম (Islam)	ইছলামী (Islami)
Buddha	বুদ্ধা (Buddha)	বুদ্ধা (Buddha)
Rajendar	ৰাজেন্দ্ৰ (Rajendar)	ৰাজেন্দ্ৰ (Rajendar)
Shudh	শুদ্ধ (Shudh)	শুদ্ধা (Shudha)
Shaleem Khan	শালিম খান (Shaleem Khan)	শালিম খান (Shaaaleem Khan)
Ram	ৰাম (Ram)	ইছলামী (Ram)
Rose	ৰোজ (Rose)	ৰোজ (Rose)
* Basab Nath	বাসবনাথ (Basab Nath)	বসাবনাথ (Bosab Nath)
* Kanyakumari	কন্যাকুমাৰী (Kanyakumari)	নায়াকুমাৰী (Nayakumari)
* Jatashankar	জাটশংকৰ (Jatashankar)	জতশংকৰ (Jotashankar)

Table 10. Sample translation by integrated system without transliteration

Input	Actual output	Predicted output
This is primarily grown in greenhouse.	ইয়াক প্ৰধানকৈ গ্ৰীণহাউচত উৎপন্ন কৰা হয়। (Yak pradhankai greenhouse utpanno kora hoy.)	ইয়াক প্ৰধানকৈ <unk> উৎপন্ন কৰা হয়। (Yak pradhankai <unk>utpanno kora hoy.)
Kamlesh is a maths teacher by profession.	কমলেশ পেছাত এজন গণিত শিক্ষক। (Kamalesh peshat ejan ganit shikshak.)	কশ এজন গণিত শিক্ষক হয়। (Kash ejan ganit shikshak hoy.)
Rangoli is made in festive occasions.	ৰংগোলী উৎসৱৰ অনুষ্ঠানত তৈয়াৰ কৰা হয়। (Rangoli utsabar anushtanat tayer karahoy.)	<unk>উৎসৱৰ অনুষ্ঠান তৈয়াৰ কৰা হয়। (<unk>utsawar anushtan tayer kara hoy.)
Geeta Kumari is a wrestler from Haryana.	গীতা কুমাৰী হাৰিয়ানাৰ এগৰাকী মল্ল যোদ্ধা। (Geeta Kumari hariyana egraki malla yoddha.)	গী হাৰিয়ানা মল্ল যোদ্ধা। (Gee hariyana egraki malla yoddha.)
Amjad Khan was a great Indian actor.	আমজাদ খান এজন মহান ভাৰতীয় অভিনেতা আছিল। (Amjad Khan ejan mahan bharati abhineta asil.)	মহান ভাৰতীয় অভিনেতা। (Khan ejan mahan bharatiyo abhineta asil.)
Bihar is one of the highly populated states in India.	বিহাৰ ভাৰতৰ এখন অতিজন বসতিপূৰ্ণ ৰাজ্য। (Vihar bharatar ekhan atijan vasatipurna rajya.)	হা ভাৰতৰ এখন অতিজনবস ৰাজ্য। (ha bharatar ekhan atijan rajya.)

model has six layers of encoders and decoders, 1536 input embeddings with 16 attention heads, and a 4096 feed-forward dimension. It uses label smoothing of 0.1 and gradient clipping of 1.0 in the Adam optimizer and starts with a learning rate of 5e-4 and 4000 warm-up steps. We tested this model using the identical test corpus as our proposed model, and the findings are presented in Table 12. In the second comparison, the Microsoft Azure Cognitive Services Translation API were used to translate all the sentences in our test set. The experimental results of the proposed integrated system and the two existing approaches are compared in terms of the BLEU

Table 11. Sample translation by integrated system with transliteration

Input	Actual output	Predicted output
This is primarily grown in greenhouse.	ইয়াক প্ৰধানকৈ গ্ৰীণহাউচ উৎপন্ন কৰা হয়। (Yak pradhankai greenhouse utpanno kora hoy.)	ইয়াক প্ৰধানকৈ গ্ৰীণহাউচত উৎপন্ন কৰা হয়।। (Yak pradhankai greenhouse utpanno kora hoy.)
Kamlesh is a maths teacher by profession.	কমলেশ পেছাত এজন গণিত শিক্ষক। (Kamlesh peshat ejan ganit shikshak.)	কমলেশ এজন গণিত শিক্ষক হয়। (Kamlesh peshat ejan ganit shikshak.)
Rangoli is made in festive occasions.	বংগোলী উৎসৱৰ অনুষ্ঠানত তৈয়াৰ কৰাহয়। (Rangoli utsabar anusathanat tayer karahoy.)	বংগলী উৎসৱৰ অনুষ্ঠান তৈয়াৰ কৰা হয়। (Rangoli utsawar anusathanat tayer karahoy.)
Geeta Kumari is a wrestler from Haryana.	গীতা কুমাৰী হাৰিয়ানৰ এগৰাকী মল্ল যোদ্ধা। (Geeta Kumari hariyana egraki malla yoddha.)	গীতা কুমাৰী হাৰিয়ানা মল্ল যোদ্ধা। (Amjad Khan ejan mahan bharati abhineta asil.)
Amjad Khan was a great Indian actor.	আমজাদ খান এজন মহান ভাৰতীয় অভিনেতা আছিল। (Amjad Khan ejan mahan bharati abhineta asil.)	আমজাদ খান এজন মহান ভাৰতীয় অভিনেতা। (Amjad Khan ejan mahan bharati abhineta.)
Bihar is one of the highly populated states in India.	বিহাৰ ভাৰতৰ এখন অতিজন বসতিপূৰ্ণ ৰাজ্য। (Vihar bharatar ekhan atijan vasatipurna rajya.)	বিহাৰ ভাৰতৰ এখন অতিজনবস ৰাজ্য। (Vihar bharatar ekhan atijanabas rajya.)

Table 12. Performance comparison between integrated system and existing models

Model	BLEU	sacreBLEU	METEOR	TER	chrF2	RIBES
Proposed integrated system	28.13	32.22	35.76	69.44	40.76	49.23
Samanatar IndicTrans model (Ramesh et al., 2022)	15.43	17.01	23.78	83.78	27.32	36.26
MSFT translation service (MSFT, 2020)	20.31	23.60	27.44	76.32	33.41	41.33

Table 13. Human evaluation scores for English to Assamese translations

Model	Adequacy	Fluency	Overall rating
Proposed integrated system	3.91	4.63	4.27
Samanatar IndicTrans model (Ramesh et al., 2022)	2.81	3.26	3.03
MSFT translation service (MSFT, 2020)	3.01	3.74	3.37

score, sacreBLEU, METEOR, and TER, and the results are presented in Table 12 and Figure 9, respectively.

According to the experimental findings shown in Table 12, it is observed that the proposed integrated model for English to Assamese translation outperforms both MSFT and Samanantar’s IndicTrans translation system. Specifically, the proposed model achieves a BLEU score of 28.13, compared to 20.31 and 11.43 for MSFT and Samanantar’s Indic translation system, respectively. Additionally, in Tables 14, 15, and 16 we presented a few sample predicted sentences to further examine the translation quality of the integrated system, MSFT, and Samanantar’s IndicTrans translation system. From the predicted sentences in Tables 13, 14, 15 and 16 it is observed that the integrated system is capable of accurately translating named entities and performs better than the existing systems for English to Assamese translation. Furthermore, the proposed integrated

Table 14. Sample output from MSFT system

Input sentence	Predicted output	Actual output
This is primarily grown in greenhouse.	এইটো মুখ্যতঃ সেউজগৃহত খেতি কৰা হয়। (ato mukhyat seujagrihat kheta kara hoy.)	ইয়াক প্ৰধানকৈ গ্ৰীণহাউচ উৎপন্ন কৰা হয়। (iyak prathamata seujagrihat utpanna kara hoy.)
I live in Pride	মই গৌৰৱত বাস কৰো (moi gouravat bas karo.)	মই প্ৰাইডত বাস কৰো (moi Pride-t bas karo.)
My name is Moon	মোৰ নাম চন্দ্ৰ (mor nam chandra.)	মোৰ নাম মুন (mor nam mun.)

Table 15. Sample output from Samanantar system

Input sentence	Predicted output	Actual output
This is primarily grown in greenhouse.	ইয়াৰ মুখ্য উৎপাদন হ’ল গ্ৰীনহাউছ। (iyar mukhy utpadon hol Greenhouse.)	ইয়াক প্ৰধানকৈ গ্ৰীণহাউচ উৎপন্ন কৰা হয়। (iyak prathamata seujagrihat utpanna kara hoy.)
I live in Pride	মই গৌৰৱত জীয়াই আছে (moi gouravat jiai acho.)	মই প্ৰাইডত বাস কৰো (moi Pride-t bas karo)
My name is Moon	মোৰ নাম চন্দ্ৰ। (mor nam chandrar.)	মোৰ নাম মুন

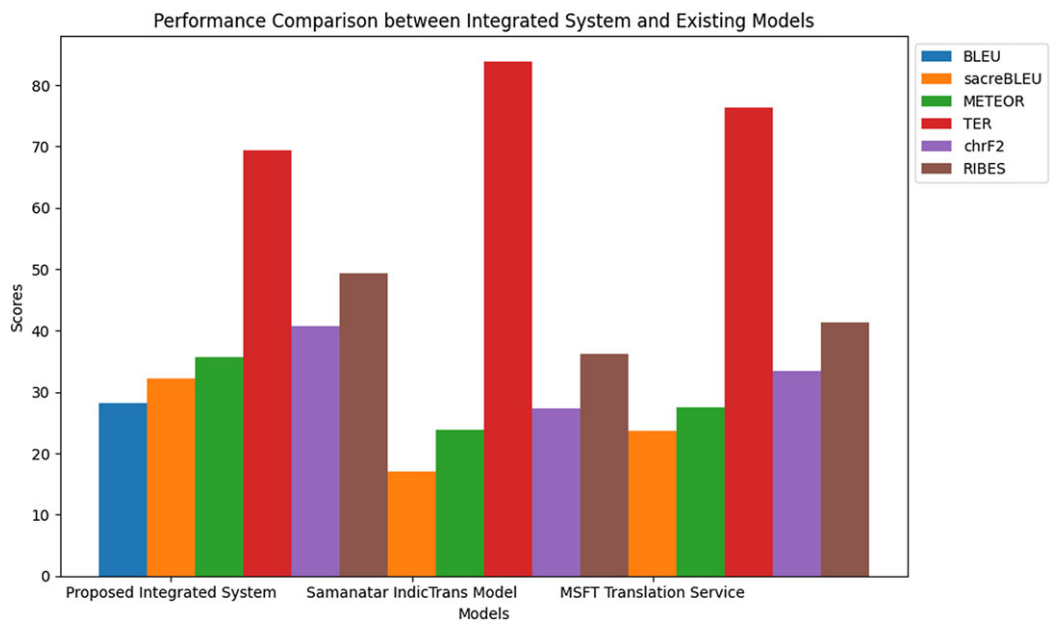


Figure 9. Bar chart for performance comparison II.

system is able to transliterate unknown named entities, as demonstrated in Table 8. Overall, the translation quality of the proposed integrated model surpasses that of the existing systems. To thoroughly assess the translation quality of MT models, we conducted a human evaluation following the approach of Laskar *et al.* (2023), in addition to using automatic metrics. This is important because it can capture nuances that automatic metrics often miss. Human evaluation method focused on two key aspects: adequacy, which measures how accurately the translation conveys the meaning of the source sentence, and fluency, which assesses how natural and idiomatic the translation is. To obtain an overall score, we averaged the adequacy and fluency ratings. Three evaluators

Table 16. Sample output from integrated system

Input sentence	Predicted output	Actual output
This is primarily grown in greenhouse.	ইয়াক প্ৰধানকৈ গ্ৰীণহাউচ উৎপন্ন কৰা হয়। (yack pradhankai greenhouse utpanno kara hoy.)	ইয়াক প্ৰধানকৈ গ্ৰীণহাউচত উৎপন্ন কৰা হয়। (iyak prathamikbhabe greenhouse-t utpanna kara hoy.)
I live in Pride	মই প্ৰাইডত বাস কৰোঁ (moi Pride-t bas karo.)	মই প্ৰাইডত বাস কৰোঁ (moi Pride-t bas karo.)
My name is Moon	মোৰ নাম মুন (mor nam mun.)	মোৰ নাম মুন (mor nam mun.)

independently assessed translation quality of three models using a 1–5 scale for a randomly chosen set of 100 sample sentences. Table 13 summarizes the human evaluation scores for the three models, providing insights into English-to-Assamese translation performance.

6. Conclusion

This paper proposed an integrated translation model for English to Assamese that combines a GRU-based translation model with a transliteration model and a hybrid attention mechanism. The integrated model is able to translate named entities accurately, leading to improved translation accuracy. Furthermore, subword tokenization in the transliteration system partially addresses the OOV issue. It achieves significantly better scores across multiple evaluation metrics, including BLEU, TER, METEOR, and sacreBLEU, when compared to two other existing translation models.

For the current work, we have performed experiments on the TDIL corpus. However, we recognize that publicly available datasets such as the Samanantar corpus (containing about 138,353 English-Assamese parallel sentences) represent valuable additional training data. In future work, we plan to incorporate the Samanantar dataset into our models as well. In the future, we also plan to explore other languages from the northeastern region of India to develop a multilingual system for translation. Additionally, to enhance the transliteration system, we suggest developing a model capable of extracting phonemes, which are the basic units of sound in a particular language, from the bilingual corpus.

**Competing interests.** The authors declare none.

References

Abbas M. R. and Asif D. K. (2020). Punjabi to ISO 15919 and Roman Transliteration with phonetic rectification. *ACM Transactions on Asian and Low-Resource Language Information Processing* 19(2), 1–20.

ai4bharat (2022). Multilingual transliteration service. Retrieved from. Available at: <https://xlit.ai4bharat.org/> (accessed 20 July 2022).

Al-Muzaini H. A., Al-Yahya T. N. and benhidour H. (2018). Automatic Arabic image captioning using RNN-LSTM-based language model and CNN. *International Journal of Advanced Computer Science and Applications* 9, 6–6.

Athavale V., Bharadwaj S., Pamecha M., Prabhu A. and Shrivastava A. (2016). Towards deep learning in Hindi NER: an approach to tackle the labeled data sparsity. *CoRR*, abs/1610.09756.

Bahdanau D., Cho K. and Bengio Y. (2014). Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations*.

Blasi D., Anastasopoulos A. and Neubig G. (2022). Systematic inequalities in language technology performance across the world’s languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1, Long Papers)*.

Baruah K., Das P., Hannan A. and Sarma S. K. (2014). Assamese-English bilingual machine translation. *International Journal on Natural Language Computing* 3(3), 73–82.

- Cheng Y.** (2019). Semi-supervised learning for neural machine translation. In *Joint Training for Neural Machine Translation. Springer Theses*. Springer, pp. 1–193.
- Cho K., Merriënboer B. V., Bahdanau D. and Bengio Y.** (2014). Learning phase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Džambazov G., Kocmi T., Rosa R. and Bojar O.** (2019). Neural transliteration for slavic languages: A case study of Ukrainian and Bulgarian. *Transactions of the Association for Computational Linguistics* 7, 145–159.
- Grundkiewicz R. and Heafield K.** (2018). Neural machine translation techniques for named entity transliteration. In *Proceedings of the Seventh Named Entities Workshop*, pp. 89–94.
- Hadj Ameur M. S., Meziane F. and Guessoum A.** (2017). Arabic machine transliteration using an attention-based encoder-decoder model. *Procedia Computer Science* 117, 287–297.
- Hassan H., Aue A., Chen C., Chowdhary V., Clark J., Federmann C., Huang X., Junczys-Dowmunt M., Lewis W. and Li M.** (2018). Achieving human parity in Automatic Chinese-to-English news translation. arXiv preprint arXiv: 1803.05567.
- He W., He Z., Wu H. and Wang H.** (2016). Improved neural machine translation with SMT features. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30(1).
- Hochreiter S. and Schmidhuber J.** (1997). Long short-term memory. *Neural Computation* 9(8), 1735–1780.
- Jankowski P., Choloniewski J., Trzciński T. and Kocoń J.** (2021). Multilingual neural machine translation with soft decoding and language-specific attention for Slavic languages. *Transactions of the Association for Computational Linguistics* 9, 1–16.
- Karimi S., Scholer F. and Turpin A.** (2011). Machine transliteration survey. *ACM Computing Surveys* 43(3), 1–46.
- Kaur A. and Goyal V.** (2018). Punjabi to English machine transliteration for proper nouns. In *2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*.
- Kaur S. and Singh C.** (2015). Conversion of Punjabi text to IPA using phonetic symbols. *International Journal for Technological Research in Engineering* 2, 3183.
- Laitonjam L. and Singh S. R.** (2022). A hybrid machine transliteration model based on multi-source encoder-decoder framework: English to manipuri. *SN Computer Science* 3(2), 125.
- Lakshmi B. S. S. and Shambhavi B. R.** (2019). *Automatic English to Kannada Back-Transliteration Using Combination-Based Approach*. Emerging Research in Electronics, Computer Science and Technology, pp. 159–170.
- Le N. T., Sadat F., Menard L. and Dinh D.** (2019). Low-resource machine transliteration using recurrent neural networks. *ACM Transactions on Asian and Low-Resource Language Information Processing* 18(2), 1–14.
- Laskar S. R., Khilji A. F. U. R., Pakray P. and Bandyopadhyay S.** (2020). EnAsCorp1.0: English–Assamese corpus. In *Proceedings of the 3rd Workshop on Technologies for MT of Low Resource Languages*, pp. 62–68.
- Laskar S. R., Khilji A. F. U. R., Pakray P. and Bandyopadhyay S.** (2022). Improved neural machine translation for low-resource english-assamese pair. *Journal of Intelligent & Fuzzy Systems* 42(5), 4727–4738.
- Laskar S. R., Paul B., Dadure P. and Manna R.** (2023). English-Assamese neural machine translation using prior alignment and pre-trained language model. *Computer Speech & Language* 82, 101524.
- Lavie A. and Denkowski M.** (2009). The METEOR metric for automatic evaluation of machine translation. *Machine Translation* 23(2-3), 105–115.
- Luong M. T., Pham H. and Manning C. D.** (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.
- Masmoudi A., Khmekhem M. E., Khrouf M. and Belguith L. H.** (2020). Transliteration of Arabizi into arabic script for Tunisian dialect. *ACM Transactions on Asian and Low-Resource Language Information Processing* 9(2), 1–21.
- Makarov I., Petrushkov V. and Biemann C.** (2020). Unsupervised transliteration mining for low-resource languages: A case study on Ukrainian. *Transactions of the Association for Computational Linguistics* 8, 459–475.
- Nath B., Sarkar S. and Das S.** (2022). *Neural Machine Translation for Indian Language Pair Using a Hybrid Attention Mechanism*. Innovations in Systems and Software Engineering.
- Papineni K., Roukos S., Ward T. and Zhu W. J.** (2002). BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*.
- Post M., Vilar D. and Rebele T.** (2018). Fast, cheap, and good!. In *Introducing SacreBLEU, a Class of Automatic Evaluation Metrics for Machine Translation*. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1875–1886.
- Popović M.** (2015). chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, Lisbon, Portugal: Association for Computational Linguistics, pp. 392–395.
- Phan-Vu H. H., Tran V. T., Nguyen V. N., Dang H. V. and Do P. T.** (2019). Neural machine translation between Vietnamese and English: An empirical study. *Journal of Computer Science and Cybernetics* 35(2), 147–166.
- Ramesh G., Doddapaneni S., Bheemaraj A., Jobanputra M., Ak R., Sharma A., Sahoo S., Diddle H., Kumar D., Pradeep N., Nagaraj A., Deepak S., Raghavan K., Kunchukuttan V., Kumar P. and Khapra M.S.** (2022). Samanantar: The largest publicly available parallel corpora collection for 11 Indic languages. *Transactions of the Association for Computational Linguistics* 10, 145–162.

- Sennrich R., Haddow B. and Birch A.** (2016). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725.
- Shao Y. and Nivre J.** (2016). Applying neural networks to English–Chinese named entity transliteration. In *Proceedings of the Sixth Named Entity Workshop*.
- Snover M., Dorr B., Schwartz R., Micciulla L. and Makhoul J.** (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pp. 223–231.
- Vathsala M. K. and Holi G.** (2020). RNN-based machine translation and transliteration for Twitter data. *International Journal of Speech Technology* 23, 587–599.
- Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L. and Polosukhin I.** (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 6000–6010.
- Yonghui W., Mike S., Zhifeng C., Quoc V. L. and Mohammad N.** (2016). Google’s neural machine translation system: bridging the gap between human and machine translation, pp. 1–23. arXiv preprint arXiv: 1609.08144.
- Younes J., Souissi E., Achour H. and Ferchichi A.** (2018). A sequence-to-sequence based approach for the double transliteration of Tunisian dialect. *Procedia Computer Science* 142, 238–245.
- Younes J., Achour H., Souissi E. and Ferchichi A.** (2020). Romanized Tunisian dialect transliteration using sequence labelling techniques. *Journal of King Saud University - Computer and Information Sciences* 32(7), 3482–3491.
- Yuchen L., Long Z., Yining W., Yang Z., Jiajun Z. and Zong C.** (2018). A comparable study on model averaging, ensembling and reranking in NMT.
- Zhou J., Cao Y., Wang X., Li P. and Xu W.** (2016). Deep recurrent models with fast-forward connections for neural machine Translation. *Transactions of the Association for Computational Linguistics* 4, 383.
- MSFT** (2020). Microsoft Cognitive Services Text Translation. Available at: <https://portal.azure.com/create/Microsoft.CognitiveServicesTextTranslation> (accessed 19 November 2022).
- TDIL-DC** (2006). Indian language technology proliferation and Development Centre - TDIL-DC: Corpus. Retrieved from. Available at: <https://tdil-dc.in/> (accessed 18 May 2020).
- Zhang M., Li H., Kumaranz A. and Banchs R. E.** (2016). Whitepaper of NEWS. 2016 shared task on transliteration generation. In *NEWS ’16 Proceedings of the 2016 Named Entities Workshop: Shared Task on Transliteration*, Berlin, Germany.