International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 10 Issue: 10 | Oct 2023 www.irjet.net p-ISSN: 2395-0072

Advancements in Hindi-English Neural Machine Translation: Leveraging LSTM with Attention Mechanism

Tribhangin Dichpally[1], Yatish Wutla[2], Uday Vallabhaneni[3], Rohith Sai Midigudla[4]

[1,2,3,4] *Student, Department of Computer Science and Engineering (SCOPE),*
VIT-AP University,
Amravathi, Andhra Pradesh, India.

Abstract - *People worldwide are brought together by language translation to work together, share knowledge, and develop relationships. To make the language translation process free from human error and less expensive, we devised an automated approach. Machine learning makes the process less time-consuming and simpler. The ability of neural machine translation achieves notable advancements over archaic methods, including Rule-Based and Statistical Machine Translation. This paper examines neural machine translation models for the Hindi-English language pair based on the Long Short-Term Memory (LSTM) model, which incorporates an attention mechanism. This paper discusses the various steps involved in building an LSTM model data preprocessing, creating encoders and decoders, and analyzing training and validation accuracies. We verify if LSTM performs better than a simple RNN network. Our model helps translate Hindi to English sentences, which can be helpful in the tourism, education, and entertainment industries. Future possibilities to enhance our approach have also been discussed.*

***Key Words*: Decoders, Encoders, Hindi, LSTM, RNN**

1.INTRODUCTION

Translation includes more than simply translating words from one language to another. It builds bridges between cultures and helps people to communicate better. However, skilled translators are needed to do this who have a good knowledge of both the languages. Often, meanings can be misinterpreted. We require a system that accurately translates between languages and is less prone to human error.

We have developed a translation model which converts text from Hindi to English. We have selected Hindi because it is the native language of around 260 million people worldwide. Not everyone in India knows English fluently, and tourists from other countries might struggle to express themselves. Hindi has become essential for organizations who wish to develop and expand their business in India. Various books and movies about our culture are written in Hindi and can be successful around the world if translated accurately.

Human translators often make mistakes, leading to adverse consequences, especially in professional situations. We require an automated approach to these issues that also produces the fewest possible errors. Also, creating a model will save time and give us results in seconds.

Hindi and English have very different structures, and a dataset containing these languages is known as a parallel dataset. To create this model, we will be using sequence-to-sequence deep learning. Since our sentences will vary in length after conversion, our project comes under the category of canonical sequence-to-sequence. We must enter the complete input sequence at once to see the output.

Most of the existing projects have used Recurrent Neural Networks, which take sequences of text as input; each cell state and output become the input of the next time step. This is the loop or recurrence present in the hidden layer. But if we have a group of sentences, we need the model to remember older information to understand and interpret the current text. RNNs have a weak memory. Hence, these models are not so accurate when we have paragraphs or pages of text to translate.

However, we have made the use of LSTM or Long-Term short-memory layers to build our model. The hidden state and cell state vectors are created by the encoder after it has read the input sequence and summarized the data. We only keep the internal states after eliminating the encoder's outputs. The decoder begins creating the output sequence using these initial states.

2. LITERATURE SURVEY

The model used in this article [1] to learn the mapping from one human language to other employs intermediate numerical encoding. The foundation of the neural

translation system is two end-to-end neural networks. Spanish word sequences are converted into an array of integers with the same meaning by the first neural network. The second neural network gains the ability to convert the numbers into a series of words that all have the same English meaning.

Both text files contained lists of sentences in the same order, one of which was an English list and the other a Spanish list. For each language, the master data files were divided into sections for training, development, and testing.

Our translation system was supported in handling words that had never been seen before by a method called sub word segmentation. Byte Pair Encoding, or BPE, is the specific implementation of sub word segmentation that is utilized. The model's general structure is as follows: At each time step, the encoder selects one element from the input sequence, examines it, gathers data about it, and propagates it forward. The intermediate vector represents the model's final internal state as it emerges from the encoder. It includes details about the entire input sequence to aid the decoder in making precise predictions. Given the entire text, the decoder predicts an output for each time step.

This study helps the user create and test a basic translation system with essential components. This helps the user comprehend the basic ideas behind the encoder-decoder model used in translation and piques their interest in more in-depth studies of the subject.

In this research [2], a parallel corpus of Assamese-English language pairs was incorporated, and two neural machine translation (NMT) systems were constructed. The first system employed a transformer model equipped with self-attention, while the second system utilized a sequence-to-sequence recurrent neural network (RNN) along with an attention mechanism. These NMT systems are denoted as NMT-1 and NMT-2, respectively. With bilingual evaluation understudy (BLEU) ratings of 10.03 for English-to-Assamese translation and 13.10 for English translation, the NMT-2 methodology performed noticeably better than earlier methods.

The source and target phrases are tokenized in the preprocessing stage, it additionally creates an indexing dictionary for words while the system is being taught. Both systems have a vocabulary dimension of 50,022 for English sentences and 46,845 for Assamese sentences. The encoders and decoders are the main components of the NMT system's architecture. The entire source sentence is compressed by the encoder into a context vector, which is then represented by the decoder. When a sentence is too long, the encoder cannot include all of the relevant details in the context vector. This led to the development of the attention-based encoder-decoder paradigm. Self-Attention System NMT System LSTM-based sequence-to-sequence RNN improves long sentence quality, but accuracy deteriorates and parallelization is constrained. The transformer model is presented as a solution to these problems. "Self-attention" refers to the attentional process that utilizes the transformer paradigm. The decoder employs a beam search algorithm to reduce the search space of potential translations for the best translation.

The model used in this article [3] to learn the mapping from one human language to another employs intermediate numerical encoding. The foundation of a neural translation system is two end-to-end neural networks. Spanish word sequences are converted into an array of integers with the same meaning by the first neural network. The second neural network gains the ability to convert the numbers into a series of words that all have the same English meaning.

The results of recent calculations have an impact on those from earlier calculations. Using this technique, neural networks may find patterns in a stream of information. For example, we may use it to infer the next word in a phrase from the first few words. We must first separate the text into sentences. Our neural network can only read one sentence at a time and will perform poorly if we attempt to feed it an entire paragraph at once. Text normalization will then be completed. We'll make sure that words are capitalized consistently inside a single phrase, fix any unusual formatting of the punctuation, and get rid of any odd quotation marks.

The model's general structure is as follows: At each time step, the encoder selects one element from the input sequence, analyzes it, gathers data about it, and propagates it forward. The intermediate vector represents the model's final internal state as it emerges from the encoder. It includes details about the entire input sequence to aid the decoder in making precise predictions. Using the entire text as input, the decoder generates a prediction for each time step.

3.PROPOSED METHODOLGY

The following subsections provide detailed coverage of the three key phases in system operations: data preprocessing, system training, and system evaluation.

3.1 About Dataset

We have used IIT Bombay English-Hindi Translation Dataset [4] from Kaggle. There are 1,561,840 occurrences of Hindi - English Translation in this database. This is a dataset of parallel corpora. The phrase "parallel corpus" refers to a corpus that comprises both the original writings in the language as well as translations into a variety of languages.

3.2 Data Preprocessing

The main objectives of the preprocessing stage are tokenizing source and target phrases and building a dictionary that indexes the terms used during training. All terms that are distinct are listed in the dictionary. To evaluate the trained system's convergence, the validation data set was taken.

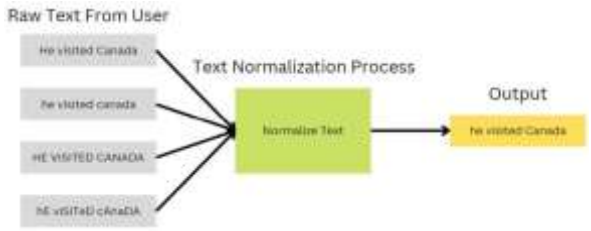

Fig -1: Preprocessing of Sentence

To make our data more uniform and straightforward for our model to analyze, we will preprocess it by changing all the letters to lowercase [5]. The text will then be cleared of any special characters and numbers. Finally, we will eliminate any unnecessary quotation marks and spaces. We should also shuffle our dataset and split it into training and testing datasets. Figure 1 shows an example of this process. After the text normalization process, we will split our dataset into training and testing datasets. Table 1 shows the number of instances in each section.

Table -1: Dataset Split

	Sentence cases
Testing	202
Training	12323

3.3 System Training

Input documents in both the source and target languages are inputted into a sequence-to-sequence recurrent neural network (RNN) with an attention mechanism. [6] after the dataset has been preprocessed, and the system is trained to predict translations. The system is trained over an epoch, which is a predetermined number of iterations. The Neural Machine Translation (NMT) system architecture's [7] primary units are the encoder and decoder. The input sequence as a whole need to be encoded into a fixed-size context vector or summary vector that contains each of the input sentence's essential parts at each input time step. This is the job of the encoder module. The two-layer LSTM component network has 500 nodes per layer that make up the unidirectional sequencer's encoder function. A sentence contains a number of contextual dependencies that must be selectively remembered or forgotten.

This is made possible by the gating system in the LSTM units. The decoder is constructed of a two-layer network of LSTM components, with 500 nodes in each layer, much as the encoder [8]. Its task is to read from the context vector while stepping through the output time steps.

Conventional encoder-decoder Neural Machine Translation systems [9] face a challenge in that they condense the source sentence into a vector of constant size. There is, therefore, a loss of important information if the sentence is too long.

The context vector's encoder doesn't contain all the necessary data. An attention mechanism is implemented to overcome this issue [10], allowing the decoder to concentrate on distinct sections of the source sequence at certain decoding phases.

Instead of relying on a fixed sentence vector, the attention mechanism employs annotation vectors for individual words, each with a predetermined length [11].

These attention weights, representing weighted combinations of the source sentence's annotation vectors, are harnessed to compute the context vector. The equation elucidates how these attention weights assess the significance of the source word relative to the generation of the target word, as demonstrated by the alignment model depicted in (1).

$$e_{i,j} = align\left(z_{i-1}, h_j\right), \forall j \in 1,2, \dots T, \forall i \in 1,2, \dots T^{'} \quad (1)$$

The alignment score associated with the jth source word that corresponds to the i-th target word is represented here by the symbol ei,j. The variable zi−1 represents the decoder's prior state, while hj denotes the annotation vector associated with the jth source word. Furthermore, T and T 0 denote the source and target phrases' relative lengths. Equation (2) describes the conversion process that the alignment scores go through in order to get the probabilistic values known as attention weights (i,j).

$$\alpha_{i,j} = \frac{\exp\left(e_{i,j}\right)}{\sum_{j'} \exp\left(e_{j',i}\right)} \quad (2)$$

In the end, as shown in equation (3), the context vector is obtained by using the annotation vector (ci).

$$c_i = \sum_{j=1}^{T} \alpha_{i,j} \times h_j \quad (3)$$

Once the context vector has been established, we proceed to compute the subsequent state of the decoder. This computation involves employing a non-linear function (in the case of LSTM) that takes into account the context vector ci, the preceding target word μi−1, and the decoder's previous state zi−1, as detailed in equation (4).

$$z_i = f(c_i, \mu_{i-1}, z_{i-1} \quad (4)$$

3.3 System Testing

After system training comes system testing and translation, where the best translations are found using beam search, an improved version of the best first search heuristic [12]. The model of a machine translation is given in Figure 2. It shows the structure of a similar project on English-to-Hindi conversion [13].

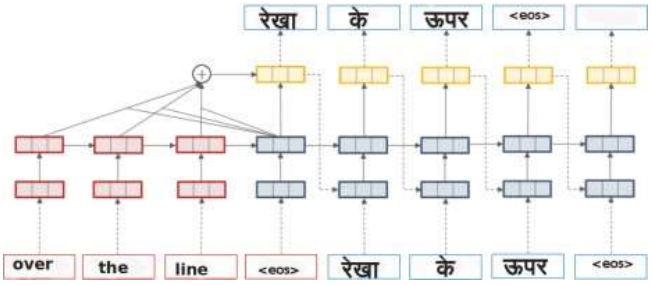

Fig -2: LSTM Model Visualization [14]

We used LSTM to avoid problems of loss of older information and to make our model more effective [15]. Long Short-Term Memory (LSTM) is the name given to a specific class of RNNs that can learn long-term dependence problems [16]. This one was found in 1997 by Schmidhuber and Hochreiter and was updated and widely circulated in that study. Each LSTM cell has a flow, as shown in Figure 3.

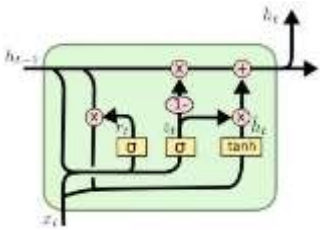

Fig -3: Flow of Lstm Cell [17]

Both the encoder and the decoder employ LSTM models. Reading the input sequence allows the encoder to compress the data into hidden state vectors. Only the internal states are retained; all other outputs are discarded. The LSTM reads each series of data in turn.. We thus state that LSTM reads the input in 'k' time steps if the input is a sequence of length 'k'. The architecture of our model depicted by the plot model function in Keras is given in Figure 4.

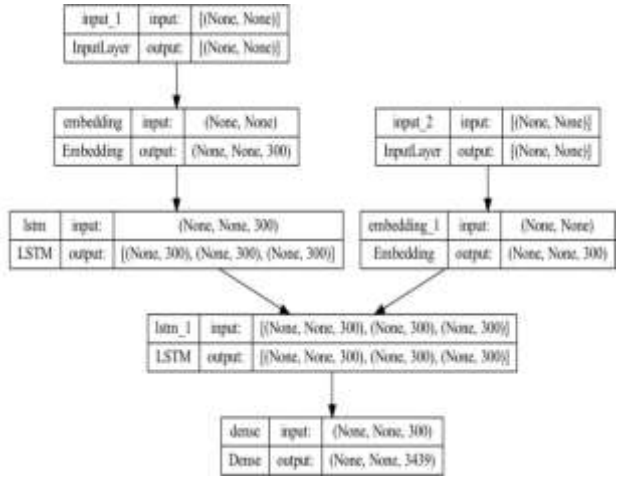

Fig -4: Proposed Model

4.RESULTS AND DISCUSSIONS

In comparison to previous RNN models, our model provided a greater accuracy. It was capable of precisely translating more than 90% of Hindi phrases into English.

The training plot pertaining to the accuracy and loss of the LSTM model elucidates the correlation between training accuracy and validation accuracy as the model undergoes successive epochs.

The depicted graph demonstrates a consistent rise in training accuracy with each additional epoch of training. However, the validation accuracy exhibits a comparatively slower increase and eventually levels off. In the plot shown in Chart 1, the training accuracy attains its zenith at 0.95 after 15 epochs. In contrast, the validation accuracy reaches a peak of 0.90 after 16 epochs.

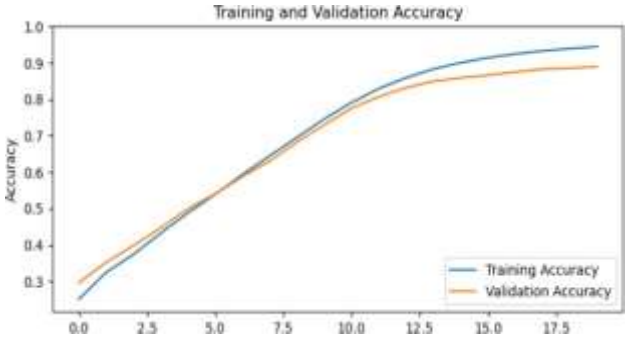

Chart -1: Model Accuracy

While accuracy metrics provide information about how well a model can produce accurate translations, they do not provide a comprehensive evaluation of translation quality. In contrast, the BLEU score accounts for both fluency and sufficiency and provides a more thorough evaluation of the model's performance. The BLEU score is from 0 to 1.

Therefore, while assessing NMT models, it is crucial to evaluate not just the BLEU score but also accuracy and loss. The BLEU score provides a comprehensive assessment of translation quality, whereas accuracy and loss metrics shed light on the model's competence in generating precise translations. These results highlight the need for future improvements to the model's ability to predict accurately for phrases that are not included in the training dataset. The translations that our model generates have a BLEU score of 0.72, which indicates that they are fairly similar to the reference translations.

5.CONCLUSION

The outcome is better in comparison to the existing RNN and GRU systems. This model's practicality exceeds that of other models. Therefore, we do not require experts to polish our translation process at each level. The computer handles it for us. With this approach, almost every sequence-to-sequence problem may be sorted. No language rules are required to be understood for this. The algorithm itself develops these rules. Other Indian languages can be converted using the same method. Since our model is fairly accurate, it might be utilized to develop English-to-Hindi translation software, which would be very useful in industries like tourism, education, or business. We aim to further develop the model in order to generate text in various tasks and forms, such as articles and captions, and to translate text in multi-way translation.

REFERENCES

[1] Tiwari, G., Sharma, A., Sahotra, A., & Kapoor, R. (2020, July). English-Hindi neural machine translation-LSTM seq2seq and ConvS2S. In 2020 International Conference on Communication and Signal Processing (ICCSP) (pp. 871-875). IEEE.

[2] Alla, L. (2020, March 27). Using LSTM to Translate French to Senegalese Local Languages.

[3] Dedes, K., Putra Utama, A. B., Wibawa, A. P., Afandi, A. N., Handayani, A. N., & Hernandez, L. (2022). Neural Machine Translation of Spanish-English Food Recipes Using LSTM. JOIV : International Journal on Informatics Visualization, 6(2), 290.

[4] https://www.kaggle.com/datasets/vaibhavkumar11/hindi-english-parallel-corpus

[5] Su, C., Huang, H., Shi, S., Jian, P., & Shi, X. (2020). Neural machine translation with Gumbel Tree-LSTM based encoder. Journal of Visual Communication and Image Representation, 71, 102811.

[6] S. Saini and V. Sahula, "Neural Machine Translation for English to Hindi," 2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP), 2018, pp. 1-6

[7] S. R. Laskar, A. Dutta, P. Pakray and S. Bandyopadhyay, "Neural Machine Translation: English to Hindi," 2019 IEEE Conference on Information and Communication Technology, 2019, pp. 1-6

[8] Md. Sanzidul Islam, Sadia Sultana Sharmin Mousumi, Sheikh Abujar, Syed Akhter Hossain, Sequence-to-sequence Bangla Sentence Generation with LSTM Recurrent Neural Networks, Procedia Computer Science, Volume 152, 2019, Pages 51-58, ISSN 1877-0509

[9] Y, Hrithick, N, Dinesh., & R, Hariharan. (2022). Machine Translation: Spanish-to-English Translation System using RNNs. International Journal for Research in Applied Science and Engineering Technology, 10(9), 1428–1434.

[10] Vathsala, M.K., & Holi, G. (2020). RNN based machine translation and transliteration for Twitter data. *International Journal of Speech Technology*, 1-6.

[11] Shakhovska, K., Dumyn, I., Kryvinska, N., & Kagita, M. (2021). An Approach for a Next-Word Prediction for Ukrainian Language. Wirel.Commun. Mob. Comput., 2021, 5886119:1-5886119:9.

[12] Gangar, K., Ruparel, H., & Lele, S. (2021). Hindi to English: Transformer-Based Neural Machine Translation. Lecture Notes in Electrical Engineering

[13] Laskar, S.R., Manna, R., Pakray, P., & Bandyopadhyay, S. (2022). Investigation of Multilingual Neural Machine Translation for Indian Languages. WAT.

[14] Aspects of Terminological and Named Entity Knowledge within Rule-Based Machine Translation Models for Under- Resourced Neural Machine Translation Scenarios. (2020)

[15] Revanuru, K., Turlapaty, K., & Rao, S. (2017). Neural Machine Translation of Indian Languages. Compute.

[16] Dewangan, S., Alva, S., Joshi, N., & Bhattacharyya, P. (2021). Experience of neural machine translation between Indian languages. Machine Translation, 35, 71-99.

[17] Sepp Hochreiter, Jürgen Schmidhuber; Long Short-Term Memory. *Neural Comput* 1997; 9 (8): 1735–1780.
doi: https://doi.org/10.1162/neco.1997.9.8.1735