IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Ingénierie des Systèmes d'Information
Vol. 26, No. 1, February, 2021, pp. 123-127
Journal homepage: http://iieta.org/journals/isi

Kannada to English Machine Translation Using Deep Neural Network

Pushpalatha Kadavigere Nagaraj*, Kshamitha Shobha Ravikumar, Mydugolam Sreenivas Kasyap, Medhini Hullumakki Srinivas Murthy, Jithin Paul

Department of Electronics & Communication, Dayananda Sagar College of Engineering, Bengaluru 560078, Karnataka, India

Corresponding Author Email: pushpalatha-ece@dayanandasagar.edu

https://doi.org/10.18280/isi.260113

Received: 9 August 2020
Accepted: 15 November 2020

Keywords:
encoder-decoder mechanism, long short term memory, natural language processing, sequence to sequence model

ABSTRACT

In this paper, we focus on the unidirectional translation of Kannada text to English text using Neural Machine Translation (NMT). From studies, we found that using Recurrent Neural Network (RNN) has been the most efficient way to perform machine translation. In this process we have used Sequence to Sequence (Seq2Seq) modelled dataset with the help of Encoder-Decoder Mechanism considering Long Short Term Memory (LSTM) as RNN unit. We have compared our result concerning to Statistical Machine Translation (SMT) and obtained a better Bi-Lingual Evaluation Study (BLEU) value, with an accuracy of 86.32%.

1. INTRODUCTION

Natural Language Processing (NLP) provides machines with the ability to understand and deduce human languages from their interpretation and acts as a connection between human language and data science [1]. Neural Machine Translation (NMT) is a machine translation technique that uses a wide range of neural networks to predict the probability of a series of words in a single integrated pattern that forms a complete sentence to reduce the language barrier [2-4].

A Deep Neural Network (DNN) is a hierarchical organization of hidden networks (layers) that connect input and output. The DNN seeks the correct mathematical manipulation to transform the input into output [5].

Machine translation has remained unexplored in the field of some native Indian languages. To understand the information and ideas expressed in a certain language, translation is necessary. This has culminated in the introduction of automated translation from regional to national or international language [6].

Kannada which is one among the Dravidian language which has rich historical literature but has a poor resource in terms of computational linguistics which becomes a difficult task due to its syntactic and semantic variance in its literature. In the field of MT, Kannada is not explored much when compared to other Indian languages. Many research works have been focused on English-South Dravidian language (Kannada /Malayalam) utilizing SMT which was meant to be a traditional approach in machine translation [7].

2. RELATED WORKS

Translation for foreign languages like German, French, Spanish was developed first when seen its development for Indian languages, a phrase-based statistical model was developed for eight languages which include Hindi, Bengali, Gujarati, Tamil, Malayalam, Telugu, Urdu and Punjabi were translated to English [8]. Further many methods were implemented and the one in current use is NMT which has shown its excellent results for translation. NMT with attention model has been implemented for multilingual translation of five Indian languages including English translation [9], six Indian languages to Hindi translation [10], Hindi to English translation [11], Hindi to Bengali translation with comparing values of MOSES statistical machine translation (SMT) and NMT [11] with different calculation metrics for comparing its performances and accuracy [12].

Kannada-English translation is much an unexplored area in terms of Machine Translation. Some of the resources suggest a Baseline SMT using MOSES for Kannada-English Translated bible corpus [13] and English-Kannada translation through rule-based MT [14]. Translation for simple sentences is done for Kannada transliterated corpus using lexicon analysis and phrase mapper to identify display boards of Karnataka government offices [15]. In this paper, we have applied the NMT method to translate Kannada Text to English Text using Encoder-Decoder mechanism with LSTM as RNN unit, without attention mechanism.

3. PROPOSED ARCHITECTURE

Figure 1 depicts the block diagram of the proposed architecture used in the MT process. Starting with input language and translated language both require a tokenizer which splits sentences into words and gives an integer value for every unique word present in the dataset. These integer values are converted into vector values which act as an input to the LSTM cell. The Yellow blocks represent the two-layered LSTM cells which together forms the encoder and decoder part of the Seq2Seq model. X_i are the input vectors to an encoder model whereas Y_i are the discarded output of the encoder similarly X_i' and Y_i' are the input and output vectors of decoder respectively. The predicted output of the decoder is taken into the SoftMax activation function which selects which one to activate based on the vector values and

represents the probability distributions of a list of potential outcomes.

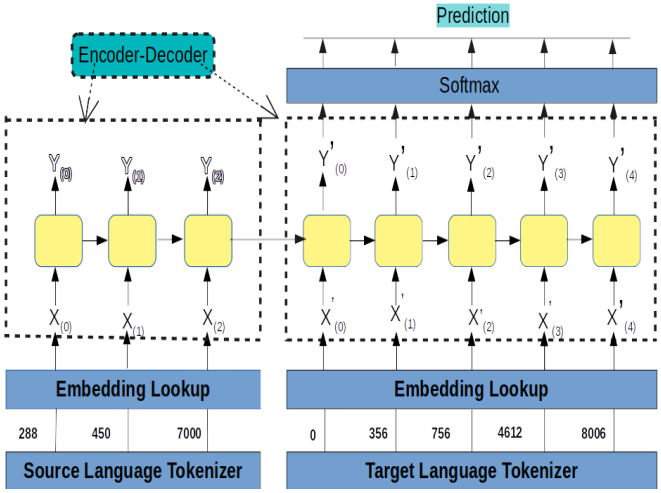

Figure 1. Proposed architecture

3.1 Sequence to Sequence Model (Seq2Seq)

A Seq2Seq model is a model that inputs a sequence of items (words, letters, time-series) and generates an output of a different sequence of items. The input is a series of word and the output is the translated series of words for a machine translation application. A typical Seq2Seq model includes two parts: an encoder and a decoder. Both are two different models of the neural networks combined into one huge network. Each of these encoders and decoders consists of series of any Recurrent Neural Network (RNN) unit, connected to work as encoder/decoder. Hence, we use LSTM as an RNN unit. These LSTM in series form the encoder and decoder part which combines to form a Seq2Seq model [16, 17].

3.2 Long short term memory (LSTM)

Long Short-Term Memory (LSTM) networks, a class of RNN is capable to process and learn to predict sequence based on its order dependencies. Hidden state (h_t) and cell state (C_t) are the two internal states present in LSTM. The information flows out through a mechanism known as cell states. So, LSTM's may remember things selectively, or forget things. An LSTM basically has four gates, to preserve and control the cell state value based on memory. They are:

- Forget gate
- Input gate
- Update gate
- Output gate

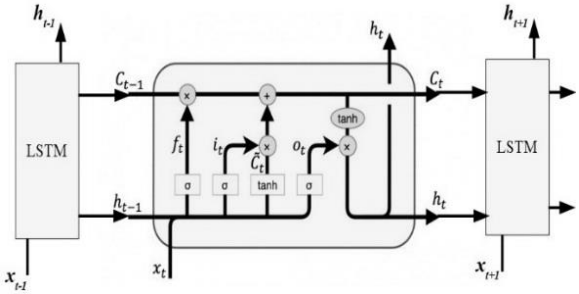

Figure 2. LSTM architecture

In Figure 2, all the gates are represented with a pointwise operator which acts as a valve to decide its flow. In LSTM, the first step is to determine which information we are seeking to discard from the cell state as it is stored in the memory previously. A sigmoid layer activates "forget gate f_t" (Eq. (1)), which performs this function. Consecutively, it is required to decide what new information must be stored in the memory of the cell state. Two elements are primarily concerned with performing this action. First, the "input gate i_t"(Eq. (2)) is a sigmoid layer which defines the values to be updated. The second element is the tanh layer $\tilde{C}_t$ (Eq. (3)), determines a new vectored candidate values which can be applied to the cell state C_t. We use "update gate C_t" (Eq. (4)) to update the old cell State C_{t-1} to produce a new cell state C_t. The final output is decided using the "Output gate o_t" (Eq. (5)). The hidden state h_t (Eq. (6)) acts as the input to the next LSTM in case of series arrangement as depicted in Figure 2. The equations for respective gates are given by:

$$f_t = \sigma(W_f . [h_{t-1}, x_t] + b_f) \quad (1)$$

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i) \quad (2)$$

$$C_t = tanh(W_C . [h_{t-1}, x_t] + b_C) \quad (3)$$

$$C_t = f_t * C_{t-1} + i_t * C_t \quad (4)$$

$$o_t = \sigma(W_O . [h_{t-1}, x_t] + b_O) \quad (5)$$

$$h_t = o_t * tanh(C_t) \quad (6)$$

3.3 Encoder-decoder mechanism

The most ideal approach for MT using Seq2Seq model is by using Encoder-Decoder architecture where both encoder and decoder consists of LSTM as its subunits.

The process starts with the encoder unit, where the LSTM's are placed in series fashion to obtain an encoder vector. Encoder vector is the value of internal states from the last LSTM of the encoder which encloses information about previous input elements. This vector will be the initial state value for first decoder LSTM which helps decoder for accurate predictions. Encoder mechanism is the same in case of training and inference process.

Decoder begins to generate output sequence using the initial state generated and initialized from the final state of encoder LSTM. During inference and training procedure, the decoder behaves differently. During training, we use teacher forcing technique to train decoder quickly as shown in Figure 3. During inference, the output from the previous time step acts as an input to the decoder at each time step as depicted in Figure 4.

Encoder summarizes all the input sequence applied to encoder into state vectors (h and C) and discards the output of the encoder. The final state vectors (h_i and C_i) of the encoder are initialized as initial input (h_0', C_0') of the decoder to generate output sequence. A decoder is just a linguistic model which depends on its initial states to generate output sequence by discarding the final state vectors.

The LSTM in both encoder and decoder sequentially reads the data sequence. Thus, if the input is a sequence of 'k' length, we ensure that the LSTM reads it in 'k' time steps.

The most important point is that the decoder's initial states (h_0', C_0') are set to the encoder's final state (h_i, C_i). This implicitly means that the decoder is trained to generate output sequences based on the encoded information by the encoder. English translated phrase should be dependent on the sentence given in Kannada for translation.

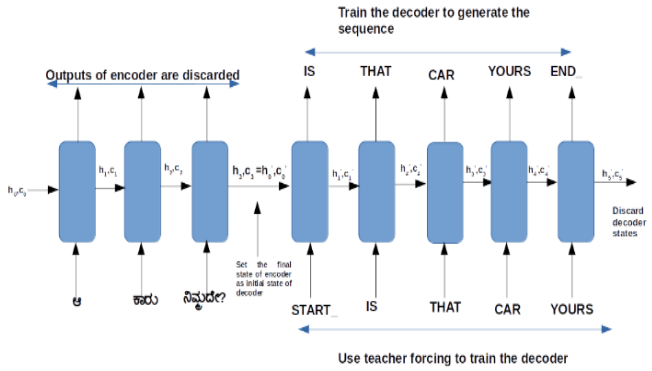

Figure 3. Encoder-decoder LSTM in training process

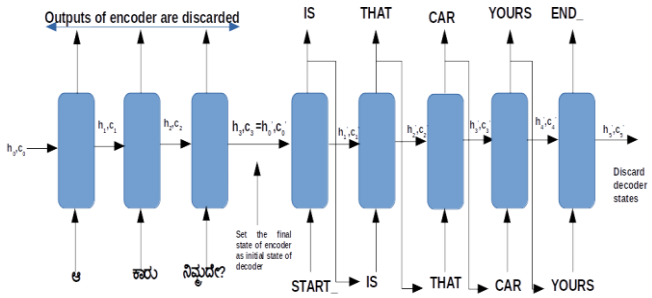

Figure 4. Encoder-decoder LSTM in inference process

During the inference process, one word will be created at a time. Therefore, the LSTM decoder is called in a loop and only a one-time step is processed every time. At each time step, we retain the decoder states and set them as initial states to the next time step i.e., the predicted output at present time step acts as input to the next time step.

4. IMPLEMENTATION

4.1 Dataset

Preparing dataset depends on the type of machine learning project. From previous researches and surveys, we have found that the corpus used textbook translation which does not influence much-spoken language. Since there is non-availability of open-source datasets for Kannada to English, hence we have developed our dataset based on the Seq-Seq model for Kannada-English in parallel format. Pre-processing of data is done to the dataset to build the models more accurately.

Table 1. English-Kannada corpus

	English	Kannada
Total no of tokens	6697	17596
No of sentences (Training)	40500	40500
No of sentences (Testing)	500	500
Max Length	15	11

A dataset which we created has 41 thousand pairs of parallel data. We split the dataset in 98:2 i.e., 98% of the dataset is for training and 2% is for testing. (40500 for training and 500 for testing the model). However, to increase the accuracy and efficiency of Translation system it is required to update the dataset frequently.

Table 1 depicts Kannada and English corpus information.

4.2 Experimental setup

According to the Corpus size, the platform in which the model must be trained will be chosen. There are various platforms to train the model in Machine Learning.

In our project, we have used Google Colab (an open-source platform) for Training purpose, where we can leverage free robust Graphics Processing Unit and Tensor Processing Unit for machine learning education and researches. We used 17GB of Tensor Processing Unit (TPU) Memory to train the project model.

For Evaluating and Predicting the sequence on a local machine, we have used 8 GB primary memory and 2 GB Nvidia 940 MX with 384 Compute Unified Device Architecture (CUDA) Cores. CUDA is developed by Nvidia as a computing platform on GPU's. CUDA helps developers speed-up compute-intensive exercises.

For Creating and Training Deep Neural Networks (DNN's), Keras with TensorFlow backend deep Learning framework is used.

5. RESULT

Bi-Lingual Evaluation Study (BLEU) score is a metric used for evaluation of predicted sentence to target sentence. It will result in 1 if there is a perfect match or 0 if there is a complete mismatch. We have got a different value while changing the weights as shown in Table 2.

Table 2. Bi-Lingual Evaluation Study (BLEU) Scores Obtained in Inference

BLEU (Weights)	Train (Scores normalized to 1)	Test (Scores normalized to 1)
BLEU-1 (1.0, 0, 0, 0)	0.642263	0.472143
BLEU-2 (0.5, 0.5, 0, 0)	0.559133	0.360877
BLEU-3 (0.3, 0.3, 0.3, 0)	0.497395	0.302902
BLEU-4 (0.25,0.25,0.25,0.25)	0.317888	0.173815

BLEU Score evaluation as in Base Line MT using MOSES tool kit [13] and NMT are compared in Table 3 represented in percentage (%). Translation time for our model was about 2-5 seconds based on the length of the input sentence. Translation time for MOSES algorithm is not available in the referred paper.

Table 3. Comparison of BLEU score for Training and Testing

Dataset	BLEU(MOSES)	BLEU(NMT)
Training set	14.5143	31.788
Testing set	10.68	17.3815

Figure 5 graph depicts Validation Loss versus Epoch. As

the number of Epochs increases, validation loss decreases till a certain epoch. The minimum loss is considered till the end of epochs. We obtained validation loss of 0.849.

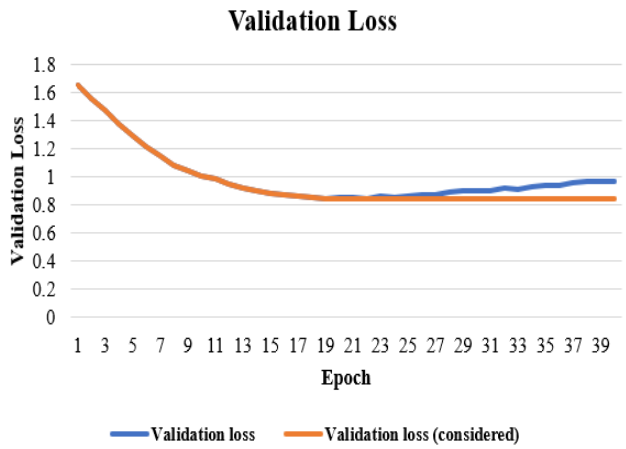

Figure 5. Validation Loss v/s Epochs Graph

Figure 6. graph represents Validation Accuracy versus Epochs. Initially, for the first epoch, validation accuracy is about 74.84%. Then as the number of epochs increases, validation accuracy starts increasing. So, the accuracy is 86.32%.

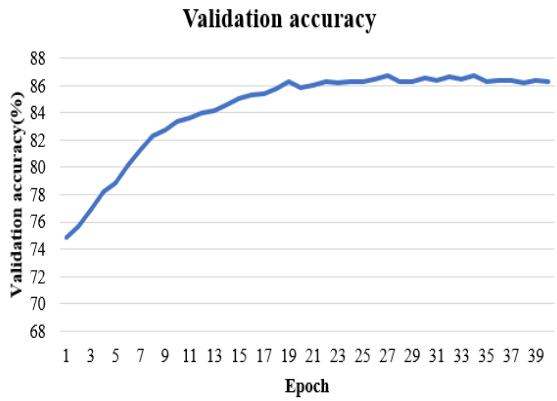

Figure 6. Validation accuracy v/s epoch graph

The real-time outputs for user inputs implemented in the above system are represented in Table 4.

Table 4. Real-time outputs obtained based on a model trained

Input sentence	ಅದು ನನಗೆ ಇಷ್ಟ
Output sentence	I like that
Input sentence	ನನಗೆ ಸಹಾಯ ಮಾಡಿ
Output sentence	help me
Input sentence	ಅವಳು ಯಾರು?
Output sentence	who's is she?

6. CONCLUSION

This paper uses Machine Translation based on the principle of Deep Neural Network (DNN) to accelerate communication among scientific workers, regardless of the language in which their findings may be expressed. LSTM works well in case of classification, processing and time series prediction, despite unknown time lags. A model is created using a dataset of 41000 data pairs and 40 epochs giving an overall loss of 0.849 and accuracy of 86.32%. The results obtained show that the model is more efficient than translation services available using Seq-Seq method.

7. FUTURE SCOPE

Gated Recurrent Unit (GRU) Layer may be used instead of LSTM which has two internal states (cell state and hidden state) whereas the internal state (hidden state) of GRU is one. It will allow the definition and description to be condensed and will also help in improved performance. Transformer architecture for time-series forecasting can be used. The parameters can be played with such as encoder and decoder layer, etc. It can improve results in tuning and training. The accuracy can be improved with training and tuning. The increase in the dataset will offer a larger base. With the larger base of data, the accuracy will also improve. Integration of automatic speech recognition (ASR) can be done to the current work. Integration of speech can be done with a large speech corpus. This can be trained to recognize the input voice samples of all pitches and frequencies and translate in real-time.

REFERENCES

[1] Otter, D.W., Medina, J.R., Kalita, J.K. (2020). A survey of the usages of deep learning for natural language processing. IEEE Transactions on Neural Networks and Learning Systems, 32(2): 604-624. https://doi.org/10.1109/TNNLS.2020.2979670

[2] Costa-Jussà, M.R. (2018). From feature to paradigm: Deep learning in machine translation. Journal of Artificial Intelligence Research, 61: 947-974. https://doi.org/10.1613/jair.1.11198

[3] Nair, L.R., Peter, S.D. (2012). Machine translation systems for Indian languages. International Journal of Computer Applications, 39(1): 24-31. https://doi.org/10.5120/4785-7014

[4] Chaudhary, J.R., Patel, A.C. (2018). Machine translation using deep learning: A survey. International Journal of Scientific Research in Science, Engineering and Technology, 4(2): 145-150.

[5] Sutskever, I., Vinyals, O., Le, Q.V. (2014). Sequence to sequence learning with neural networks. NIPS'14: Proceedings of the 27th International Conference on Neural Information Processing Systems, 2: 3104-3112.

[6] Basmatkar, P., Holani, H., Kaushal, S. (2019). Survey on Neural Machine Translation for the multilingual translation system. 3rd International Conference on Computing Methodologies and Communication (Erode, India), pp. 443-448. https://doi.org/10.1109/ICCMC.2019.8819788

[7] Unnikrishnan, P., Antony, P.J., Dr Soman, K.P. (2010). A novel approach for English to south Dravidian language statistical machine translation system. International Journal on Computer Science and Engineering, 2(8): 2749-2759.

[8] Khan, N.J., Anwar, W., Durrani, N. (2017). Machine

Translation Approaches and Survey for Indian Languages. ArXiv abs/1701.04290.
[9] Revanuru, K., Turlapaty, K., Rao, S. (2017). Neural machine translation of Indian languages. Compute 2017: 10th Annual ACM India Conference (Bhopal, India), pp. 11-20. https://doi.org/10.1145/3140107.3140111
[10] Verma, C., Singh, A., Seal, S., Singh, V., Mathur, I. (2019). Hindi-English neural machine translation using attention model. International Journal of Scientific and Technology Research, 8(11): 2710-2714.
[11] Das, A., Yerra, P., Kumar, K., Sarkar, S. (2016). A study of attention-based neural machine translation model on Indian languages. Proceedings of the 6th Workshop on South and Southeast Asian Natural Language Processing (Osaka, Japan), pp. 163-172.
[12] Shah, P., Bakrola, V. (2019). Neural machine translation system of Indic languages - An attention-based approach. 2nd International Conference on Advanced Computational and Communication Paradigms (Gangtok, India), pp. 1-5. https://doi.org/10.1109/ICACCP.2019.8882969
[13] Shivakumar, K.M., Nayana, S., Supriya, T. (2015). A study of Kannada to English baseline statistical machine translation system. International Journal of Applied Engineering Research, 10(55): 4161-4166.
[14] Reddy, M.V., Hanumanthappa, M. (2013). Indic language machine translation tool: English to Kannada/Telugu. Proceeding of 100th Science Congress (Kolkata, India), 213: 35-49. https://doi.org/10.1007/978-81-322-1143-3_4
[15] Kodabagi, M.M., Angadi, S.A. (2016). A methodology for machine translation of simple sentences from Kannada to the English language. 2nd International Conference on Contemporary Computing and informatics (Noida, India), pp. 237-241. https://doi.org/10.1109/IC3I.2016.7917967
[16] Saini, S., Sahula, V. (2018). Neural machine translation for English to Hindi. Fourth International Conference on Information Retrieval and Knowledge Management (Malaysia), pp. 1-6. https://doi.org/10.1109/INFRKM.2018.8464781
[17] Verma, A.A., Bhattacharyya, P. (2017). Literature survey: Neural machine translation. CFILT, Indian Institute of Technology Bombay, India.