Encoder Decoder Enabled Improvised Word Level English-Marathi Machine Translation

Amber Bhanarkar
Department of Computer Science and Engineering
Indian Institute of Information Technology
Nagpur, India
Email: amberbhanarkar@gmail.com

Rutvik Page
Department of Computer Science and Engineering
Indian Institute of Information Technology
Nagpur, India
Email: rutvikpage1999@gmail.com

Pooja Jain
Department of Computer Science and Engineering
Indian Institute of Information Technology
Nagpur, India
Email: iiitngpcse@gmail.com

***Abstract*—We present a novel method for the translation of English language into the Marathi language using Neural Machine Translation on a word level. We use the encoder decoder architectures of LSTMs for the encoding of the input English text into state vectors upon which further we use decoders for inference and training purposes making this a multi-model approach to neural machine translation. While there have been a number of efforts for the machine translation of a number of indigenous Indian languages into English and vice versa, there still exists a dearth of attention to the automation of translation of English language into Marathi. However, out of those which exist, most of them concentrate on Statistical Machine Translation approaches in contrast to which we present a Neural Machine Translation approach using LSTMs.**

***KEYWORDS*- Neural Machine Translation, Natural Language Processing, Neural Networks, LSTM, Indigenous Machine Translation**

I. Introduction

Natural Language Processing [1] [2] (NLP) is a popular Machine Learning [3] [4] Technique that uses Machine Learning Algorithms and their variants to solve a gamut of problems like Machine Translation [5] [6], Anaphora Resolution [7], design of chat bots and many allied problems which present a very promising solution for many important problems. In this paper, we concentrate our efforts on Machine Translation, an important method in Natural Language Processing that presents methods and algorithms to translate text from an input language to the target language using data driven approaches. Particularly, we concentrate on translation of texts from English language to the Marathi language thus proposing only a unidirectional translation i.e. we propose a novel method to convert a given English text to a corresponding text in Marathi.

Marathi is an Indian language spoken mostly in the state of Maharashtra and parts of Karnataka. The idea of Marathi being a language of communication dates back to about 1400 years (around 600AD). It belongs to the set of scheduled languages in the Eighth Schedule of the Indian Constitution. According to the 2011 census of India, the country has 6.86% or 83.1 million people who speak Marathi as first, second or third language and is the 10^{th} most spoken language in the world. Additionally, it is the third highest most spoken language in India after Hindi and Bangla.[1]. While there is a preponderance of tools available to convert English text to Hindi (and vice versa) there is a dearth of such tools when it comes to the translation of English text to Marathi. There has been a resounding research in the field of Indigenous Machine Translation but to the best of our knowledge, there are only some sources that make efforts to provide English to Marathi translation and vice versa. Google translate [2], Bing Translate [3] and Yandex translate [4] provide excellent examples to this end. Marathi differs on the most fundamental levels with the English language, i.e. the alphabets in both the languages are different. While there are 5 vowels and 21 consonants in the English language there are 14 vowels and 35 consonants in the Marathi language, which of course substantiates the fact that there are more fundamentally available sounds in Marathi, which makes tasks like speech recognition difficult due to lower language portability. Moreover, the possibility of formation of words using half consonants makes the input space relatively larger as and when compared to the English language. Though these nuances matter when the translation is being done 'to' Marathi, these form an important part in the design of training and inference models in our approach. A brief discussion of our approach has been given in the following paragraph.

Recurrent neural networks [8] [9] [10], more precisely LSTM [11] [12] or Gated Recurrent Units provide us with a very effective method to deal complex sequence problems for a large amount of data. They have quite a lot of applications in Time Series Forecasting and different problems lying in the domain of Natural Language Processing. We use neural machine translation proposed in 2013 by N.Kalchbrenner and P. Bunsom [13] in order to carry out our English to corresponding Marathi text translation using the Sequence to Sequence models which are a special class of Recurrent Neural Networks typically used to solve problems like Machine Translation and sundries. We specifically use the Encoder Decoder architecture which most commonly are LSTM models where the encoder takes the input sequence and the set of decoders provide corresponding training and inference engine mechanisms. On the most shallow level, the encoder takes an input and passes its states to the decoder which trains on the states and also provides inference on the supplied test data.

II. Paper Layout

In the following section, we point to some previous work on Machine translation in the context of Indian languages,

[1] wikipedia.org/marathilanguage

[2] translate.google.com

[3] bing.com/translator

[4] translate.yandex.com

especially Marathi and the methods used therewith. Further, we outline some preliminaries that we use in our paper following which we present our algorithm and document our findings providing some visualizations to showcase the working of the algorithm. We conclude by pointing towards some directions that could possibly improve the model further and help build more accurate and precise models not restricted only to one form of Marathi but also to different regional dialects.

III. Related Work

There has been a huge upsurge of research into the field of indigenous Indian machine language translation from European languages, especially from English. We refer the reader to [14] [15] popular surveys on machine translation and linguistic translators which show the recent advancements on the topic. Apart from the efforts that have taken in the machine translation of European languages, there also has been some progress in the arena of translation of Arabic machine translation that happens to be an important language in West Asian region [16]. A multifarious approach to machine translation has been take by Johnson et al. [17] where they enable multi-lingual machine translation using elegant single model rather than the previously proposed multi-model approach for multi-language translation i.e. an enabling of zero-shot translation. Recent works in machine translation support the fact that multi model approaches to Machine Translation between a pair of languages lead us to better results as and when compared to individual models. Sachdeva et al. [18] describe a statistical machine translation model to translate from Hindi to English using phrase based and hierarchical models and enhance over both these baselines using a regression model. Jadhav [19] presents a fresh approach for neural machine translation from Marathi to English with near perfect corpus. They train and compare a variety of neural machine Marathi to English language translators trained with BERT-tokenizer using different transformer based architectures and huggingface on well known platforms provided for sequential data analysis. [20] provide an approach for testing the effectiveness of Neural Machine Translation in the context of translation of English to Indian languages like Hindi, Tamil and Punjabi. While there has been a sizeable amount of research on machine translation in the context of Indian languages using statistical machine translation [18] [21], but here we produce one of the first approaches for solving this problem using Neural Machine Translation. Moreover, as mentioned above, there has been not a lot of research in the arena of machine translation when it comes to translation of English to Marathi and here we present one approach to provide the solution to this problem. We present an approach using encoder decoder architectures of LSTMs which we use in order to take English text as an input, train and infer upon the fed data and finally, document our results after we establish the knowledge of LSTMs and the framework that we use in order to solve the problem of the translation of English to Marathi.

IV. Preliminaries

Recurrent neural networks with Long Short Term Memory (LSTM) have emerged as an effective and scalable model for several learning problems related to sequential data [22]. LSTMs (Fig 1) have been found to be very effective in solving complex sequence related problems given a large amount of data. They have real time applications in speech recognition, time series forecasting etc. Here we give a basic overview of the functioning of LSTMs and the architecture of Sequence to Sequence model (Seq2Seq) i.e. the Encoder-Decoder architecture used for machine translation. A Sequence to Sequence model aims to map a fixed length input with a fixed length output where the length of input and output may differ.

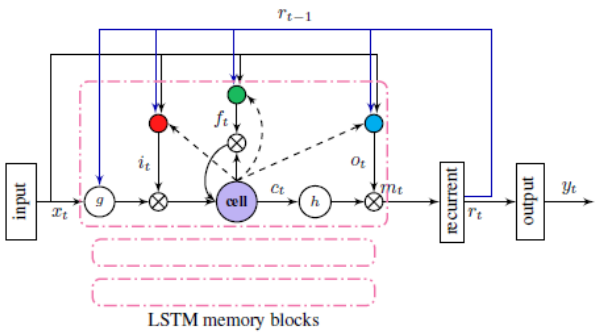

Figure 1. LSTM RNN architecture. Single memory cell shown for clarity.

Basic Neural Networks have problem of maintaining long term temporal dependencies. LSTMs on the other hand contains a memory cell which can maintain its state over time and non-linear gating units which regulate the information flow into and out of the cell. Initial version of LSTMs [23] contained only cells (possibly multiple), input and output gates.
Lately, forget gates [12] were introduced which enabled the LSTM to maintain its own gate. The gate remembers values over arbitrary time intervals and it lets reset the memory. In order to learn precise timings and not using arbitrary intervals, Peephole connections [24] were introduced so that the cell can control the gates. Peephole connections were added to make precise timings easier to learn.

An LSTM network computes a mapping from an input sequence

$$x = (x_1, ..., x_T)$$

to an output sequence

$$y = (y_1, ..., y_T)$$

by calculating network unit activation using following equations from t=1 to T:

$$i_t = \sigma(W_{ix}x_t + W_{im}m_{t-1} + W_{ic}c_{t-1} + b_i)$$

$$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1} + W_{fc}c_{t-1} + b_f)$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes g(W_{cx}x_t + W_{cm}m_{t-1} + b_c)$$

$$o_t = \sigma(W_{ox}x_t + W_{om}m_{t-1} + W_{oc}c_t + b_o)$$

$$m_t = o_t \otimes h(c_t)$$

$$y_t = \phi(W_y m_t + b_y)$$

where W denotes weight matrices (e.g. W_{ix} is the matrix of weights from input gate to the input), W_{ic}, W_{fc}, W_{oc} are diagonal weight matrices for peephole connections, b denotes bias vectors (b_i is the input gate bias vector), σ is the logistic sigmoid function, and i, f, o and c are respectively the input gate, forget gate, output gate and cell activation vectors. $\otimes$ is element wise product of vectors, g and h are cell input and cell output activation functions.

The next we introduce the Encoder-Decoder architecture (Fig 2). The most common architecture to build Seq2Seq model is the Encoder-Decoder architecture. Both the encoder and decoder are LSTM models. The encoder reads the input sentences which are sequences of words and summarizes information in internal state vectors. In case of LSTM, these are called hidden state and cell state vectors. The outputs of

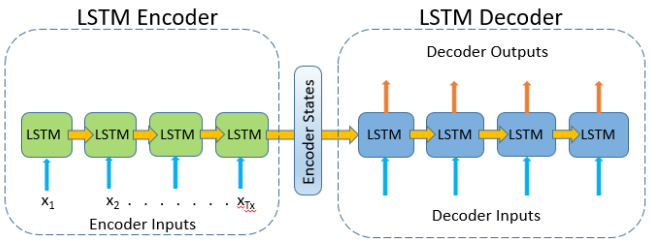

Figure 2. Systematic architecture of Encoder-Decoder model. The encoder states are intermediate vectors.

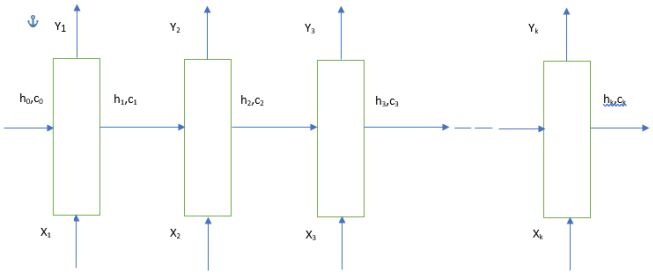

Figure 3. LSTM processing an input sequence of length k

the encoder is discarded only to preserve the internal states. The information collected from the encoder LSTM is propagated forward to the intermediate vector which acts as the final internal state produced from the encoder LSTM. It contains information about the entire input sequence to help decode LSTM make accurate predictions. The decoder on the other hand is an LSTM model whose initial states are initialized as per the output or final states of the intermediate vector. The Decoder LSTM uses these initial states to generate output sequences. Inherently, the encoder summarizes the input sequence into state vectors (precisely intermediate vectors) which are fed into decoder which generates output sequence.

V. Proposed Approach

Overall approach has been implemented in a sequence of three primary steps:

1) Encoder LSTM
2) Decoder LSTM- *Training mode*
3) Decoder LSTM- *Inference mode*

We first start with the main components of Encoder LSTM. The LSTM reads input sequences one at a time. If there are k inputs, then LSTM will read it in k time steps. The 3 main components of LSTM from Figure 3 are

1) X_i - Input sequence at time step i
2) h_i and c_i - The internal states of LSTM. h is hidden state and c is cell state.
3) Y_i - Output sequence at time step i. This is discarded in our approach.
   Consider the input sentence in English, *'Rahul is a good boy'* The corresponding output sentence in Marathi will be as mentioned in Figure 4.

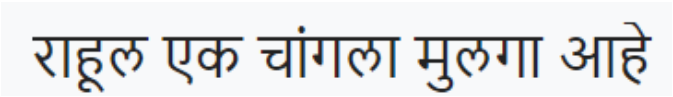

Figure 4. Marathi version of 'Rahul is a nice boy'

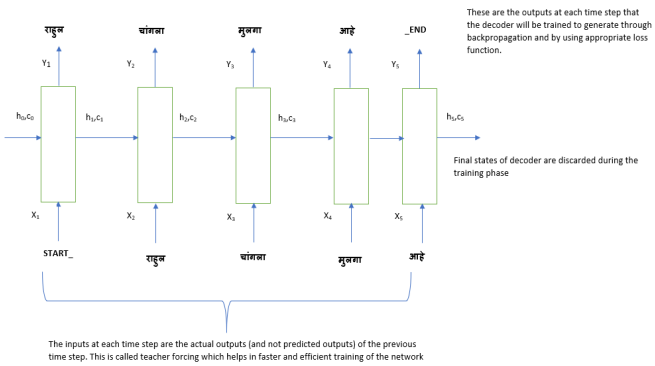

Figure 5. Decoder LSTM - Training mode

There are two techniques to break a sentence. Either by each word or by each character. We will be breaking the sentences by word and hence will call it Word Level Neural Machine Translation. The words are sent to X_i and X_i takes each of these words as input. Thus, X_1= 'Rahul, X_2= 'is', X_3= 'a', X_4= 'good', X_5= 'boy'. The LSTM will read this sentence word by word in 5 time steps. To represent each X_i as a vector, we use the technique of Embedding layer from Keras API which maps a word into a fixed length vector.
After X_i reads i^{th} word, it is then passed to the internal states, namely h_i and c_i. These states will remember what has been read till now. It is a summary of information till i^{th} time step. The vectors coming out at the last time step is called the *Thought Vector* as they summarize the entire sequence in a vector form. We have discarded the output Y_i because we will generate the output sentence i.e. equivalent Marathi sentence only after we read the entire English sentence.
The second part in our approach is the Decoder LSTM in training mode. After the encoder LSTM scanned the input sequence word by word, the decoder LSTM will generate the output sequence word by word. We have added two tokens **START_** and **END_** so that ambiguities in reading a sentence is removed. Taking the same English sentence *'Rahul is a good boy'*, its corresponding representation in Decoder LSTM is shown in Figure 5. The initial states (h0, c0) of the decoder are set to the final states of the encoder. This intuitively means that the decoder is trained to start generating the output sequence depending on the information encoded by the encoder. The translated Marathi sentence must depend on the given English sentence.
We use a technique called **"Teacher Forcing"** wherein the input at each time step is given as the actual output (and not the predicted output) from the previous time step. Finally the loss is calculated on the predicted outputs from each time step and the errors are back propagated through time in order to update the parameters of the network.

The last part in our approach is the Decoder LSTM in inference mode. This is the part where we test our model with translation. The algorithm for inference mechanism goes as
1) During inference, one word is generated at a time. Thus, the decoder LSTM is called in a loop, every time processing only one time step.

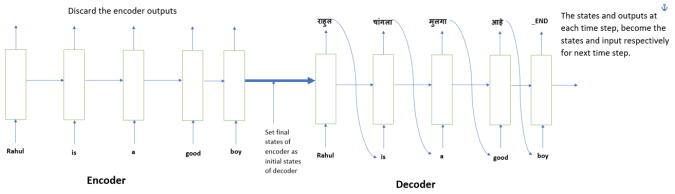

Figure 6. Summary of inference process

2) The initial states of the decoder are set to the final state of the encoder.
3) The initial state of the decoder is always the *START_* token.
4) At each time step, we preserve the states of the decoder and set them as initial states for the next time step.
5) At each time step, the predicted output is fed as input in the next time step.
6) We break the loop when the decoder predicts the *END_* token.
The entire inference procedure is summarized in Figure 6

VI. Experiment and Result

We used an Encoder-Decoder model for machine translation purpose. We divided our corpus in 90:10 ratio for training and testing phase. Our corpus consisted of 38696 bilingual translated sentences, out of which 34826 were used in training phase and 3870 were used in testing phase. We used a validation set up within the training phase with 0.1 of training data. For optimization purposes *Adam optimizer* was used and *categorical cross entropy loss function* was used. We ran our training phase for 50 epochs with a batch size of 128. We trained our model on Tesla K80 TPU with 12GB RAM and 1.87 TFlops.
After training the corpus for 50 epochs, the training accuracy came to be **65.03** while the validation accuracy was **49.16**. A plot of training versus validation loss has been put up for each epoch (Figure 7). We can observe a sudden drop in the validation loss after the 30^{th} epoch. The closeness between the training loss and validation loss for the initial epochs indicates that the model was neither under-fitting not over-fitting. But observing the epochs after 40, it can be seen that validation loss starts increasing. Thus, it can be said that model begins to over-fit after 40 epochs. A plot of the training and validation accuracy is also plotted (Figure 8). Although, training and validation accuracy increases at same rate for the initial 10 epochs, divergence can be observed after 20 epochs. The training accuracy is nearly a perfect straight line, which indicates that the model will converge and attain a good training accuracy as training epochs increases.
From the training phase, we have taken two snippets of English to Marathi translations. The first snippet (Figure 9) almost accurately translates the source sentence in English to Marathi sentence. But from the second snippet (Figure 10), we can observe there is ambiguity in deciding gender of the person from source language. As gender in input English sentence is not mentioned, the predicted translation translates taking gender as male whereas the actual translation should have happened taking female as gender. After the training phase, we validated the corpus and performed testing on it. The BLEU score we got in testing phase was 0.62 and as shown in the snippet (Figure 11), the translation is near perfect. Thus from the corpus we get a validation accuracy of 49.16 and a BLEU score of 0.62. We tabulate our results for every epoch in the training as well as the validation phases in the tables. The training accuracy

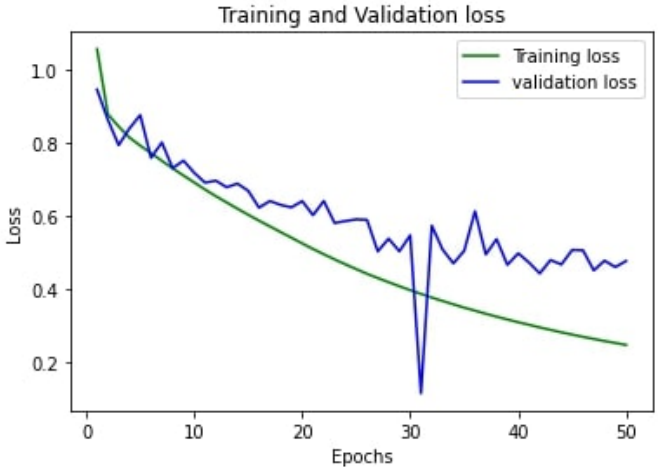

Figure 7. Training vs Validation loss for 50 epochs

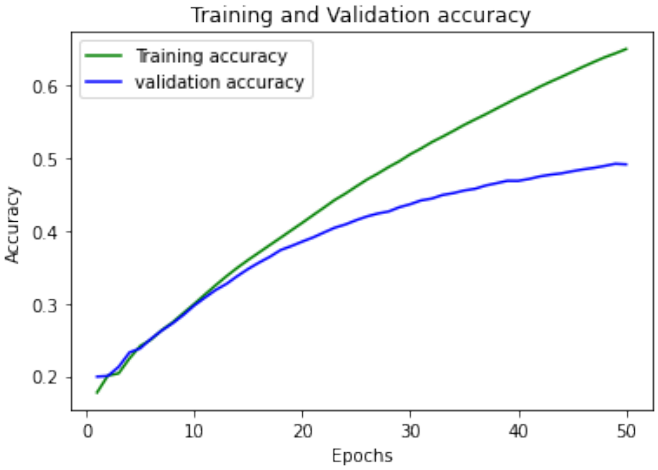

Figure 8. Training vs Validation accuracy for 50 epochs

```
[ ] k+=1
    (input_seq, actual_output), _ = next(train_gen)
    decoded_sentence = decode_sequence(input_seq)
    print('Input English sentence:', X_train[k:k+1].values[0])
    print('Actual Marathi Translation:', y_train[k:k+1].values[0][6:-4])
    print('Predicted Marathi Translation:', decoded_sentence[:-4])

Input English sentence: they want to meet you
Actual Marathi Translation:  त्यांना तुम्हाला भेटायचं आहे
Predicted Marathi Translation:  त्यांना तुला भेटायचं आहे
```

Figure 9. Snippet of a translation from training phase

```
[ ] k+=1
    (input_seq, actual_output), _ = next(train_gen)
    decoded_sentence = decode_sequence(input_seq)
    print('Input English sentence:', X_train[k:k+1].values[0])
    print('Actual Marathi Translation:', y_train[k:k+1].values[0][6:-4])
    print('Predicted Marathi Translation:', decoded_sentence[:-4])

Input English sentence: were you really in boston
Actual Marathi Translation:  तू खरच बॉस्टनमध्ये होतीस का
Predicted Marathi Translation:  तू खरच बॉस्टनमध्ये होतास का
```

Figure 10. Snippet of a translation from training phase showing ambiguity between gender translation

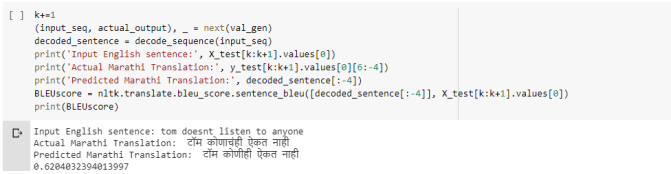

Figure 11. Snippet of a translation from testing phase with BLEU score

and losses (categorical cross-entropy) are tabulated in the table 1. As mentioned, we split the corpus in the ratio 90:10, which shall act as a reference for the evaluation and contrast of the validation and training tables.

Table I
Accuracy and Categorical Cross-Entropic Loss In Training Phase

Training Phase		
Epochs	Accuracy	Loss
10	0.2996	0.6914
20	0.4112	0.5253
30	0.5054	0.3980
35	0.5459	0.3504
40	0.5838	0.3106
45	0.6188	0.2768
50	0.6503	0.2486

We mention the validation phase statistics (specifically Validation Phase accuracy and validation phase loss) in table 2. As mentioned above, the validation phase reveals a little bit of ambiguities in gender identification and the identification of different pronoun constructs, which is of course a problem given some fundamental differences in the grammatical constructs of the languages in question.

Table II
Accuracy and Categorical Cross-Entropic Loss In Validation Phase

Training Phase		
Epochs	Accuracy	Loss
10	0.2974	0.7178
20	0.3853	0.6408
30	0.4368	0.5476
35	0.4556	0.5050
40	0.4689	0.4985
45	0.4821	0.5074
50	0.4916	0.4777

VII. Conclusion And Future Work

In this paper, we presented a novel approach to the translation of one of the most spoken languages in the world, i.e. English to one of the most widely spoken Indian languages Marathi using Neural Machine Translation. We proposed a model that initiates an excitement about the field with high enough accuracy and a near perfect BLEU score of 0.62. We presented the variation of the accuracy and losses over different epochs and also infer which areas present over fitting and under fitting. We now present what more could be done with the model in order to further the accuracy of the model. We reckon that the use of attention in LSTMs could help improve the accuracy of the system as has happened with the translation of English to many other Indian languages. Also, dropouts and other forms of regularization techniques can be used to mitigate over-fitting. Moreover, we would also like to draw readers' attention to tune different parameters used in the model, which could be done by the use of higher computing power. Furthermore, we would also like to assert the fact that a more voluminous data set with diverse sentences shall be better with these deep learning models.

Acknowledgment

The authors would like to thank their institute where they are currently enrolled in an undergraduate course in computer science and engineering. We would also like to thank our course coordinator for Natural Language Processing, Dr. Pooja Jain for valuable guidance and opportunity to carry out investigation in the given domain. Furthermore, the authors would also like to thank all anonymous reviewers for their helpful remarks.

References

[1] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[2] R. Ponnusamy, "A systematic survey of natural language processing (nlp) approaches in different systems," vol. 4, 01 2018.

[3] O. Obulesu, M. Mahendra, and M. ThrilokReddy, "Machine learning techniques and tools: A survey," in *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2018, pp. 605–611.

[4] R. Behera and K. Das, "A survey on machine learning: Concept, algorithms and applications," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 2, 02 2017.

[5] R. Dabre, C. Chu, and A. Kunchukuttan, "A survey of multilingual neural machine translation," *CoRR*, vol. abs/1905.05395, 2019. [Online]. Available: http://arxiv.org/abs/1905.05395

[6] A. P. J., "Machine translation approaches and survey for Indian languages," in *International Journal of Computational Linguistics & Chinese Language Processing, Volume 18, Number 1, March 2013*, Mar. 2013. [Online]. Available: https://www.aclweb.org/anthology/O13-2003

[7] R. Sukthanker, S. Poria, E. Cambria, and R. Thirunavukarasu, "Anaphora and coreference resolution: A review," *CoRR*, vol. abs/1805.11824, 2018. [Online]. Available: http://arxiv.org/abs/1805.11824

[8] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[9] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[10] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[11] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[12] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," 1999.

[13] N. Kalchbrenner and P. Blunsom, "Recurrent continuous translation models," *EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, vol. 3, pp. 1700–1709, 01 2013.

[14] B. J. Dorr, P. W. Jordan, and J. W. Benoit, "A survey of current paradigms in machine translation," ser. Advances in Computers, M. V. Zelkowitz, Ed. Elsevier, 1999, vol. 49, pp. 1 – 68. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S006524580860282X

[15] J. Slocum, "A survey of machine translation: its history, current status, and future prospects," *Computational linguistics*, vol. 11, no. 1, pp. 1–17, 1985.

[16] A. Alqudsi, N. Omar, and K. Shaker, "Arabic machine translation: a survey," *Artificial Intelligence Review*, vol. 42, no. 4, pp. 549–572, 2014.

[17] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado, M. Hughes, and J. Dean, "Google's multilingual neural machine translation system: Enabling zero-shot translation," Google, Tech. Rep., 2016. [Online]. Available: https://arxiv.org/abs/1611.04558

[18] K. Sachdeva, R. Srivastava, S. Jain, and D. Sharma, "Hindi to English machine translation: Using effective selection in multi-model SMT," in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland: European Language Resources Association (ELRA), May 2014, pp. 1807–1811.

[19] S. Ashok Jadhav, "Marathi To English Neural Machine Translation With Near Perfect Corpus And Transformers," *arXiv e-prints*, p. arXiv:2002.11643, Feb. 2020.

[20] A. Pathak and P. Pakray, "Neural machine translation for indian languages," *Journal of Intelligent Systems*, vol. 28, no. 3, pp. 465–477, 2019.

[21] B. U. Khan Jadoon N., Anwar W., "Statistical machine translation of indian languages : a survey," vol. 31, pp. 2455–2467, 2019.

[22] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," 1995.

[24] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," Tech. Rep., 2000.