# Analyses and Modeling of Neural Machine Translation for English-to-Khasi

**Mark S Nonghuloo, Nagaraja Rao A**

***Abstract:*** ***Language barrier is a common issue faced by humans who move from one community or group to another. Statistical machine translation has enabled us to solve this issue to a certain extent, by formulating models to translate text from one language to another. Statistical machine translation has come a long way but they have their limitations in terms of translating words that belongs to an entirely different context that is not available in the training dataset. This has paved way for neural Machine Translation (NMT), a deep learning approach in solving sequence to sequence translation. Khasi is a language popularly spoken in Meghalaya, a north-east state in India. Its wide and unexplored. In this paper we will discuss about the modeling and analyzing of a NMT base model and a NMT model using Attention mechanism for English to Khasi.***

***Keywords:*** ***Deep Learning, Recurrent Neural Network, LSTM, Neural Machine Translation, Semi-Supervised Machine Learning***

## I. INTRODUCTION

Humans have been writing down text and speaking a variety of languages for thousands of years. Our brain has understood the meaning of each word from reading and speaking a language by grasping the emotions from them. Natural Language Processing (NLP) is a branch of Artificial Intelligence (AI) that helps computers to comprehend the complexity of human languages. Computers cannot perceive what the human language is trying to convey, but recent advances in Machine Learning have enabled us to create models that help us bridge the gap between machines and human languages. There are many factors as to why NLP is difficult and one of them is because of the vagueness of natural languages i.e. a word can have a different meaning in different contexts. For example, the word “bank” has two different meanings in “the bank of a river” and “a bank loan”. While it is normal for humans to differentiate the two words from its meaning to the context, word sense disambiguation (WSD) [1] is a difficult task for computers.
Word embeddings are a comparatively new field of study proposed by Turain (Turian et al., 2010). He proposed a mechanism to convert words into numeric arrays that inherently tune themselves to values that map all the words in an N-dimensional space. This representation can be optimized to include disambiguation to very effective levels. Further deep learning was used to improve on this representation [7]. Thus using this combination of Word Embedding and Deep Neural Networks we plan to obtain complex pattern mappings of the English Language. Machine translation is a method of converting a sequence of characters from one language into another. In generating text from one language to another there is no one best single output, since the human language is flexible and vague. For a machine to predict text or convert text from one language to another language is a very challenging task. NMT is an approach to solve all the problems of the machine translation system. The main benefit of using an NMT is the ability to learn directly and map the input text to the associated output text from one end to the other (Yonghui, Mike, Zhifeng, Quoc, and Mohammed, 2016). Recurrent Neural Network being a strong and high-dimensional model with non-linear dynamics helps them to remember information from the past. In the past 20 years, research on RNN was quite less when compared to the other neural networks. Martens (Martens, 2010) developed an improved Hessian-Free optimizer (HF) which was powerful enough to train very deep neural networks. By combining RNN and HF optimizer (Ilya, James, and Geoffery, 2011) large RNN were trained with the help of an HF optimizer for predicting the next character in a stream of text.

## II. BACKGROUND STUDY

### A. Artificial Neural Networks

Neural networks are well known for their ability to extract meaning from complex or inaccurate data, they can detect trends and extract patterns that are impossible to be noticed by the human eye or standard computers algorithms. A trained neural network has the ability to learn how to do tasks based on the data provided, it can also organize and represent information it receives while learning and has the capability to compute a problem in parallel. Neural networks are popularly addressed as an artificial neural network (ANN). ANN has a similar neural structure to that of the human brain; it consists of highly interconnected nodes called neurons. An ANN can be considered as a mathematical model that process information These neurons work in parallel to compute information or solve a specific problem. Each node is a simplified model of a real neuron that receives the input signal from other nodes and sends the output signal to other nodes. [4]

**Revised Manuscript Received on June 22, 2020.**
***** Correspondence Author
**Mark S. Nonghuloo**, Department of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu, India. E-mail: marksheridan.n2018@vitstudent.ac.in
**Nagaraja Rao A.**, Department of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu, India. E-mail: nagarajaraoa@vit.ac.in

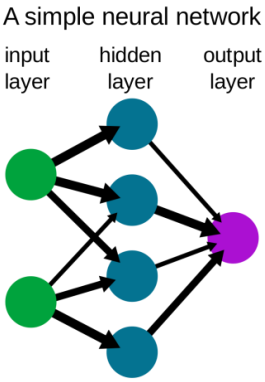


**Figure 1: Structure of an Artificial Neural Network[4]**

The neurons of the human brain are clustered together in three-dimensional space. This conveys that the process in a neural network is dynamic, self-organizing, and interactive. The neurons in an ANN can only be represented in two dimensional and can have a limited number of layers. The structure of an ANN comprises three main layers, the input layer, hidden layer, and the output layer. The input layer receives data from input files. The output layer returns the computed values to the real world. The hidden layer which is the main layer of a neural network consists of neuron and each neuron is a function to solve a specific problem.

### B. Recurrent Neural Network:

The human brain cannot simply think of a solution from scratch, it needed to learn from past experience and hold on to that memory to form a solution. The thoughts of a human are persistent. Recurrent neural networks are similar to how the human brain works. It is an interconnection of neurons with loops in them, allowing them to hold information. The ability of the RNN to remember is what makes it unique in computing sequential data by learning from its past input to produce an output. A recurrent neural network can be considered as repetitive copies of the same network, where each layer carries information to the next.

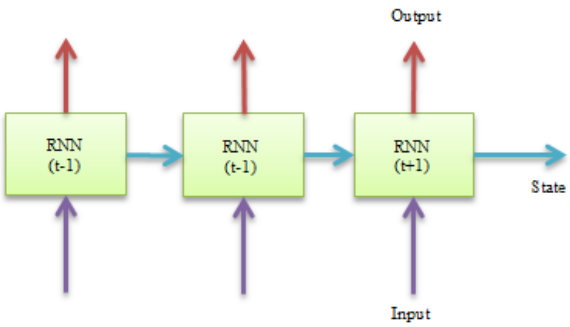


**Figure 2: Recurrent Neural Network Architecture. Instead of a fixed number of nodes, the architecture allows flexibility over time**

An RNN is a feed-forward network used to model sequential data or lists. At each step, the RNN receives an input, updates its hidden state, and predicts an output. The hidden state of an RNN integrates this information over many timesteps and makes an accurate prediction. RNN uses backpropagation to update the weights of each neuron. The gradient of an RNN is easy to compute using backpropagation, but the gradient can be very unstable since

### C. Long Short-Term Memory

The main problem of a Recurrent neural network is its inability to hold information from a long sequence input. Long Short-term Memory (LSTM) networks are basically RNNs with a memory. LSTM allows RNN to remember information over a long period of time. The three main operations of an LSTM is to read, write and delete information from its own memory. This LSTM memory is seen as a gated cell, where the cell decides whether to delete or store the information.

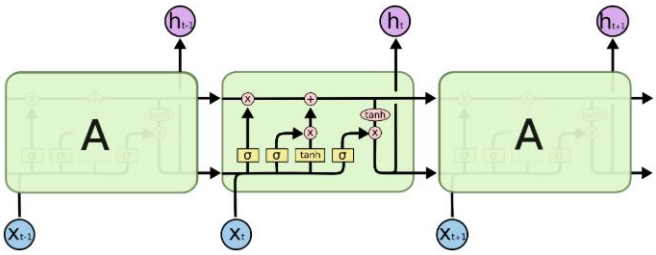


**Figure 3: Architecture of Long Short-Term Memory[11]**

An LSTM consists of three gates: input, forget, and output gate. These gates control the entry of new input, the removal of information if it is not important and the output at the current time step. The gates contain sigmoid activation, meaning that they range from 0 to 1. This makes backpropagation easier, because the multiplication of any number by 0 is 0, resulting in the update to forget the information, and the multiplication of any number by 1 is the value of the number it, therefore the value stays. LSTM solves the vanishing gradient problem present in RNNs by keeping the gradient steep enough.

### D. Neural Machine Translation

The architecture of a machine translator consists of two LSTM networks, an encoder, and a decoder. Encoder and Decoder machine translation outperforms the traditional statistical machine translation. Encoder-Decoder architecture has become a standard approach for neural machine translation. The key benefits of this model are the capability to train a single end-to-end model on the input sentences and output sentences and the ability to handle source and target sequences of any length. Encoder: The job of the encoder is to read the input sequence of text and represent it into an internal state vector. The input is processed into the RNN, this results in a hidden state that encodes each word with its left context. To get the right context we run another RNN from right to left. Since we have two RNN that are running in the opposite direction to each other, this network is called a bidirectional RNN. The outputs are discarded and only the state vectors are preserved. The state vectors are then fed to the decoder to produce the output sequence. Decoder: The decoder takes the initial state from the encoder and starts generating output sequence. In each time step the initial inputs are the actual outputs of the previous time step. This method is called

teacher forcing which helps in faster and more efficient training of the network.

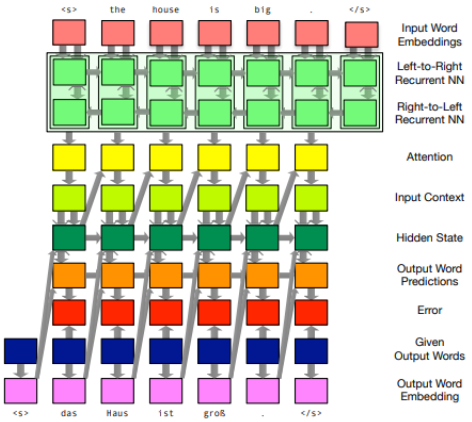


**Figure 4: Architecture of a neural machine translation model[10]**

### E. NMT using Attention

One of the limitations of a seq2seq Neural Machine Translation is that it is unable to process long input sentences because it computes the value of the hidden state for each and every input but sends only last the computed information from the Encoder to the Decoder. So instead of passing only one vector representation of the sentence, using Attention, we pass the vector representation of every word on the input sentence.

Attention was introduced by Dzmitry Bahdanau, as a solution to improve the sequence-to-sequence model in machine translation. The steps were taken by Attention to are as follows:

a) Hidden states are generated by the Encoder for each element in the input sentence
b) Alignment scores are computed between the decoder's hidden state and the encoder's bidden state
c) Use softmax on the Alignment score
d) Multiply the respective alignment score and their encoder hidden state to form a context vector
e) Decoding the output

## III. PROPOSED SYSTEM

In this part of the paper we propose the algorithms and techniques to be used. It consists of the base NMT model and the methods being used at each step.

We used word vector representation which is basically a vector of weights where every element is encoded to a vector with a word in the vocabulary.

A RNN can be used for such a problem but RNN has its drawbacks, RNN cannot hold information for a long time hence the need of another neural network that can hold information began and LSTM was the solution. The LSTM has a separate memory cell to hold information in it. Another reason why we chose LSTM model was because it solves the vanishing gradient problem, the gates that are present in the LSTM are the main features that help solve this problem. LSTM are basically RNNs but with a memory, which they can read, write and delete information to and from it.

The gates of an LSTM handles functions like the decision to store information, when to allow reads, writes, and discard. The gates are analog, which uses an element-wise multiplication by sigmoid,. The main key benefit of using analog over digital is being differentiable, and therefore makes it is more appropriate for backpropagation.

The gates are similar to the ANN's node, they perform tasks on the signals they receive, the strength of the signal decides whether they pass on or discard information and import, which they filter out with their own sets of weights. These weights can be modified by the learning process to yield better output. Through this procedure of decision making, adjusting weights via gradient descent, and backpropagation error, the cells learn whenever data is allowed to enter, leave, or be deleted.

## IV. IMPLEMENTATION

To proceed with the implementation of the paper, an efficient model and a large dataset is needed. But since the Khasi language has limited data online and offline, we self compiled the data from various sources (totoeba.org, etc) and created a bilingual text which is suitable for training our model. The dataset contains of 10,987 sentence pair in English and Khasi.

The data we have is raw data and requires data clean up. We need processing because the raw data has punctuations, some text contain upper and lower case, special characters and duplicate words. Firstly, we need to clean the data and split it.

To preserve the special characters in Khasi we loaded and cleaned the data by considering the Unicode characters. In each line of the data, there are pairs of sequence, in English and Khasi separated by tab character. We split each line from the data and then each phrase from a line. We cleaned the data by removing non-printable characters, punctuation characters, tokens that are non-alphabetic and normalizing UNICODE to ASCII, and all uppercase characters to lowercase. The cleaned dataset consist of 10,000 pairs and it is now saved for training. The length of the pairs increases as we go through the dataset. We reduced the number of pairs mainly because the length of the pairs is too long and the size of the word vector will be too huge.

We use a tokenizer for both the English sequence and the Khasi sequence separately, basically to map words to integers. Each input and output must tokenize with the help of a tokenizer and padded with 0s to its maximum length. We do this because we want to perform word embedding on the sequences and one hot encode on the output sequence. We use a one hot encoded because we need the model to predict the probability of each word in the vocabulary as the output. We used an encoder-decoder LSTM model, the first-end model called the encoder, encodes the input and then decode sequence by sequence by the backend model called the decoder. The parameters for the model such as size of the source and number of distinct words in the output, the number of memory units , and the maximum length of sequence can be specify.

We also used an encoder-decoder LSTM with Attention, using 10 attention

layers. We train the model with batch size of 32 and 64 examples for 50 and 100 epochs, and hidden units of 512 units. The training was done using an RTX 2080Ti.

## V. RESULTS AND OBSERVATIONS

The training took around 5 hours for the model without attention and around 3 hours for the model with attention. For 50 epochs the model does not completely translate the sentence correctly. But the model using attention displayed much promising results compared to the other model.

**Table 1: BLEU Score of Models trained for 50 epochs**

| Batch Size | Baseline NMT | NMT with Attention |
|---|---|---|
| 32 | 6.012547 | 10.589124 |
| 64 | 6.214769 | 10.196114 |

**Table 2: BLEU Score of Models trained for 100 epochs**

| Batch Size | Baseline NMT | NMT with Attention |
|---|---|---|
| 32 | 11.015314 | 19.552926 |
| 64 | 13.933271 | 22.506252 |

This is the only BLEU score being monitored till date regarding in the field of neural machine translation for English-Khasi translation.

These are not state of the art scores, and this is because Khasi being a language with a small vocabulary size, it utilizes a single word in more than one context. For e.g. the word “bnai” which means “moon”, but it can also mean “month”. There are multiple words like this one that are being used across the language. Another observation is the combination of two words to create a single word, this is very common in the Khasi language. For e.g. the word “bha” which means “good”, and the word “briew” which means “human”, if we add them up we get the word “bhabriew ”, which means “pretty/handsome”.

## VI. CONCLUSIONS AND FUTURE WORKS

From the observations, we have collected and the results projected, we will either need to fine-tune the model based on the structure of the Khasi Language or a very large amount of data is required to overcome all the issues that are present in the current language model. The collection of bilingual data is something that should be taken seriously especially for a language like Khasi that has almost no source of data, which also has potential for more research in the field of NLP.

## REFERENCES


1. Hwee Tou Ng and Hian Beng Lee. 1996. *Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach*. In Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics, pages 40–47
2. Rie Kubota Ando and Tong Zhang. 2005. *A framework for learning predictive structures from multiple tasks and unlabeled data*. Journal of Machine Learning Research, 6:1817–1853
3. Rie Kubota Ando. 2006. *Applying alternating structure optimization to word sense disambiguation*. In Proceedings of the Tenth Conference on Computational Natural Language Learning, pages 77–84.
4. Sonali. B. Maind, Priyanka Wankar,*Research Paper on Basic of Artificial Neural Network*, International Journal on Recent and Innovation Trends in Computing and Communication, 2014, Vol 2, Issue 1
5. Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio, *Word representations: A simple and general method for semi-supervised learning*. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics,. 2010, pages 384–394
6. Martens, J. Deep learning via Hessian-free optimization. In Proceedings of the 27th International Conference on Machine Learning (ICML). ICML 2010, 2010.
7. R. Collobert and J. Weston. *A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning*. In International Conference on Machine Learning, ICML, 2008.
8. Andriy Mnih and Geoffrey E. Hinton, *A scalable hierarchical distributed language model*. In Advances in Neural Information Processing Systems 21, . 2010, pages 1081–1088.
9. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. *Efficient estimation of word representations in vector space*. In Proceedings of Workshop at International Conference on Learning Representations, 2013a.
10. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean, Distributed representations of words and phrases and their compositionality. In Advances in Neural Information Processing Systems 26,2013b, pages 3111–3119
11. Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, 2016.