Machine Translation using Deep Learning: An Overview

Shashi Pal Singh[*1], Ajai Kumar[*2], Hemant Darbari[*3], Lenali Singh[*4] Anshika Rastogi [#4], Shikha Jain [#5]
[*]AAI, Center for development of Advanced Computing, Pune, India
[*1] shashis@cdac.in , [*2] ajai@cdac.in, [*3] darbari@cdac.in, [*4]lenali@cdac.in
[#]Banasthali Vidyapith, Banasthali, Rajasthan, India
[#4] anshikarastogi1992@gmail.com , [#5] shikhaj959@gmail.com

***Abstract*: This Paper reveals the information about Deep Neural Network (DNN) and concept of deep learning in field of natural language processing i.e. machine translation. Now day's DNN is playing major role in machine leaning technics .Recursive recurrent neural network (R^2NN) is a best technic for machine learning. It is the combination of recurrent neural network and recursive neural network (such as Recursive auto encoder). This paper presents how to train the recurrent neural network for reordering for source to target language by using Semi-supervised learning methods.Word2vec tool is required to generate word vectors of source language and Auto encoder helps us in reconstruction of the vectors for target language in tree structure. Results of word2vec play an important role in word alignment of the input vectors. RNN structure is very complicated and to train the large data file on word2vec is also a time-consuming task. Hence, a powerful hardware support (GPU) is required. GPU improves the system performance by decreasing training time period.**

Keywords: Neural Network(NN), Deep neural network(DNN), convolutional neural network(CNN), feed-forward neural network(FNN), recurrent neural network(RNN), recursive auto-encoder(RAE), Long Short-term memory(LSTM).

I. INTRODUCTION

Deep Learning is a recently used approach for machine translation. Unlike the traditional machine translation, the neural machine translation is a better choice for more accurate translation and it also provides better performance. DNN can be used to improve traditional systems in order to make them more efficient.

Different deep learning techniques and libraries are requiring for developing a better machine translation system. RNN, LSTMs etc. are used to train the system which will convert the sentence from source language to target language. Adapting the suitable networks and deep learning strategies is a good choice because it tuned the system towards maximizing the accuracy of the translation system as compare to others.

A. Machine Translation

Machine translation is a method to convert the source sentence from one natural language to other natural language with the help of computerized systems and human assistance is not necessary.

Different approaches are available to create such type of systems but we require a more robust technique to create better system than existing systems. A well-trained network leads the system towards its goal, which is to generate more efficient translation system that is capable in providing good accuracy [8] [10].

B. Deep Learning

Deep learning is a new technique, widely use in different machine learning applications. It enables the system to learn like a human and to improve the efficiency with training. Deep learning methods have the capability of feature representation by using supervised/unsupervised learning; even there exist higher and more abstract layers. Deep learning currently used in image applications, big data analyses, speech recognition, machine translation etc. [8].

C. Deep Neural Networks

Neural networks with more than one hidden layer are known as deep neural networks (DNNs). These networks first enter into the training phase then implemented to solve the problem. The structure and DNNs process of training depend upon the given task.

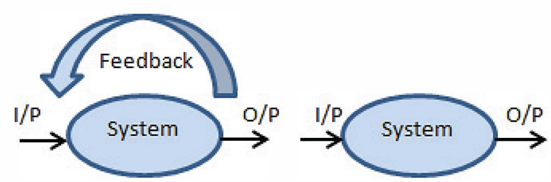


Fig. 1. Training and implementation of neural networks

II. DEEP LEARNING IN MACHINE TRANSLATION

Deep learning attracts researchers for using it in machine translation. The main idea behind this is to develop a system that works as translator. With the help of history and past experiences, a trained deep neural network translates the sentences without using large database of rules.

Machine translation consists some other related processes like word alignment, reordering rules, language modeling etc. Each process in text processing has appropriate DNN solutions as shown in the table 1 [5].

TABLE.1 DNN IN MACHINE TRANSLATION

Text Processing	**DNN Solutions**		
Word Alignment	FNN	RNN	
Translation Rule Selection	FNN	RAE	CNN
Reordering and Structure Prediction	FNN	RAE	CNN
Language Model	RAE	Recurrent NN (LSTM ,GRU)	Recursive NN
Joint Translation Prediction	FNN	RNN	CNN

III. DNN IN TRANSLATION PROCESS

After preprocessing (sentence segmentation, tokenization etc.), translation process starts with word alignment followed by reordering and language modelling.

A. Word Alignment

In word alignment input to the system is parallel sentence pair and the output is pair of words which are most related to each other. Suppose, we have source sentence $S=s_1, s_2\ldots, s_n$ and target sentence $T= t_1, t_2 \ldots t_{n'}$, then A is the set that denotes the correspondence of words between bilingual sentences, $A= \{(i, j), 1<=i<=n, 1<=j<=n'\}$
Here, (i, j) denotes the pair (s_i, t_j) which are translation of each other.

Feed forward neural network (FNN) can be used for word alignment task but it has been proven that recurrent neural network (RNN) is better choice as it maintains the history and predicts accurate next alignment on the bases of previous history of alignments (A_x based on previous history of alignments A_1^{x-1}) [5].

we want to translate source text which consists words, symbols, characters etc. A code or strategy is requiring to convert words in vector form and that conversion is based on words feature in that text.

Word embedding is key concept used in deep learning for finding the vector value of words. Word embedding is a continuous space vector representation and it has capability to capture the semantic and syntactic feature of corresponding word. Large corpus is necessary for training, it can capture information which is necessary for translation purpose. The word vector is used as an input of deep neural network. A popular tool word2vec is available to generate the vector [5].

Various models (CBOW, Skip-gram) and algorithms (Hierarchical softmax, negative sampling) work behind in word2vec processing. Word2vec reduce the dimensionality of word with the help of dimension reduction technique.

Now each vector represented by fixed-dimension vector in continuous space. If a word vector is known, then we can easily find out all the vector of the other words which are situated in same dimensions [21].

Let us take an example, where V represent the corresponding value of the word [as represented by equation (1)].

$$V[play] = V[coming] + V[come] - V[playing] \qquad (1)$$

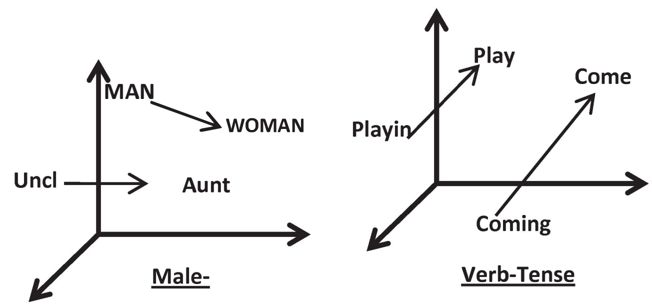


Fig. 2. Word Representation in Continuous Space

We can use the word2vec in machine translation to locate the vectors of words in corpus. If we have English-Hindi training dataset then result should be for which, we use a shallow neural network to generate the vectors and an appropriate DNN to learn these alignments. Fig.3 visualize the vector representation more clearly [21].

We can easily find out the similarity among the words with the help of dot product of their vector values [22]. The cosine similarity can be calculated as

$$cos\theta = \frac{\vec{u}.\vec{v}}{||\vec{u}||\ ||\vec{v}||}$$

Fig. 3. Word Representation

Similarity is very useful concept in case of rare words. Suppose we have an alignment A for pair (q, p), q belongs to S and p belongs to r, we want to find the correspond word of r in target language then find the nearest/similar word of r in S and find out the most suitable word s in target T such that alignment A' is generated as (r, s), here, s is the required word in target language [22].

RNN implementations for word alignment task not only learns the bilingual word embedding but also acquire the similarity between words and use the wide contextual information very effectively.

B. Rule Selection and Reordering

Once alignment process is done, translation process leads to rule selection/extraction phase. Here, rules are selected/extracted on the basis of word alignment and then reordering model is trained by word aligned bilingual text. There is a problem in choosing right target phrase/word due to language sparseness. Source sentence may have different meanings. If we have a rule R-> (S_1, …., a, T_1, …., b) then it first employed to vector representation and then similarity score is calculated to select the most suitable rule.

FNN can be used to optimize the score which leads towards better translation but bilingual constrained recursive auto-encoder outperform in this task because it tries to minimize the reconstruction error and minimize the semantic distance. The recursive auto-encoder is trained with reordering examples that are already generated from word-aligned bilingual sentences. RAE is capable enough to capture knowledge of phrase's word order information [5].

Next step is reordering and predict the structure of sentence. Combination of recursive neural network and recurrent neural network (R2NN) is a good idea to execute this. Two main concern here are 1) which two candidates composed first, 2) in which order they would be composed. To work with tree structure, recursive neural network is the best choice but if we use RNN with it then they integrate their capabilities as RNN will maintain the history that will be useful for language modelling and recursive neural network will be useful to generate tree structure in bottom-up fashion. Semi-supervised learning is used for training. R2NN is a nonlinear combination [13].

C. Language Modelling

FNN can be used to learn this model in continuous space. In this model, concatenation of word vectors is fed to input and hidden layer to find the probability of T_n based on T_1^{n-1} [5].

Recurrent neural network can be designed for language modelling because it performs very good in sequence to sequence learning task. Here we give the sequence of inputs (s_1, …, s_n) and on the basis of the sequence, it will predict sequence of output (t_1, …., $t_{n'}$). Input vectors entered to the network one by one, concatenate with previous history at hidden layers and then output is calculated at each step [9].

RNN computation can be explain by the following equations

$$h_n = \mathrm{sigm}\,(W^{hs} s_n + W^{hh} h_{n-1}) \qquad (3)$$

$$T_n = W^{th} h_n \qquad (4)$$

Two RNNs are required, one for encoding and another for decoding process. If (S, T) be the source and target sentence pair then s_1, s_2, … s_n =Encoder (s_1, s_2, …., s_n) by using chain rule, condition probability can be calculated as

$$P(S|T) = P\,(T|s_1, s_2, \ldots, s_n) \qquad (5)$$

Decoder is the combination of recurrent neural network and softmax layer [17].

It is difficult to train RNN due to long term dependences. LSTM networks avoid the problems occurred with RNN. It uses back propagation through time algorithm to learn the model parameters. [9] [4]

D. Joint Translation

Joint language and translation model is used to predict the target word with the help of unbounded history of source and target words. RNN is the best network for this. FNN and CNN only concern with the learning using networks but RNN maintains the sequence whether translation is generated left to right or right to left [14].

IV. METHODOLOGY

It is difficult to train RNN for Word Alignment so an alternative can be used in the form of bilingual corpus. We have created English- Hindi bilingual corpus that contain 1, 20,000 words with their feature values. Hence, we can fetch Hindi meaning of given word and can assign vector values to it, based on its feature [as in Figure 6]. That is, vector of English word and its corresponding Hindi word will be same and after word alignment we can proceed for further processing [as in Table 3].

Binary tree structure for source sentence is shown in Fig. 4.

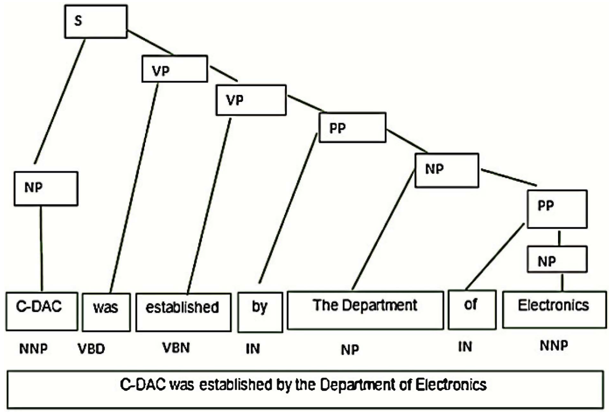
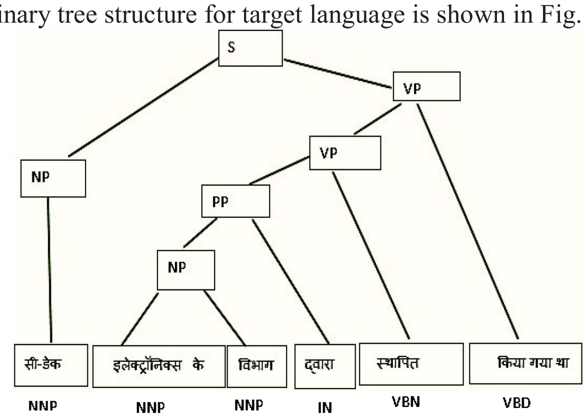


Fig. 4. Tagging and Parsing of English Sentence

Binary tree structure for target language is shown in Fig.



5.

Fig. 5. Tagging and Parsing of Hindi Sentences

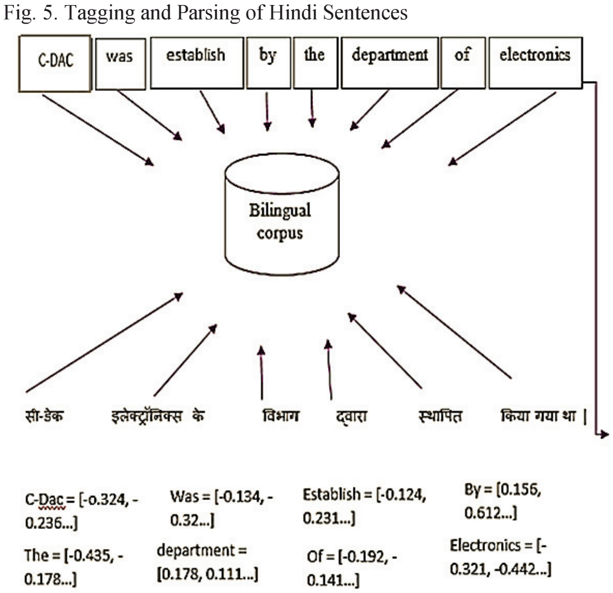


Fig. 6. Extract Information from Data

TABLE. 2 DATABASE TABLE

English	Hindi	Vectors
C-DAC	सी-डेक	[0.123, 0.107…]
Was	था	[-0.043, 0.0105…]
Established	स्थापित किया गया था	[-0.0123, 0.143 ..]
By	द्वारा	[-0.172, -0.231…]
the department	विभाग	[-0.124, -0.342…]
Of	के	[-0.442, -0.342…]
Electronics	इलेक्ट्रॉनिक्स	[-0.334, -0.344…]

TABLE. 3 RULES FOR WORD ALIGNMENT AND REORDERING

Rules for Word Alignment	R1	[C-DAC, सी डेक]
	R2	[was, किया गया था]
	R3	[of, के]
	R4	[electronics, इलेक्ट्रॉनिक्स]
	R5	[the department, विभाग]
	R6	[by, द्वारा]
	R7	[established, स्थापित]
Rules for Reordering	R8	<R7, R6> Invert
	R9	[R8, R5] Straight
	R10	[R9, R4] Straight
	R11	[R10, R3] Straight
	R12	[R11, R2] Straight
	R13	<R1, R12> Invert

Since the vector of "C-DAC" is [0.123, 0.107…] and the vector of "was" is [-0.043, 0.0105…] then we denote "C-DAC was" as parent and can find the vector of this phrase by concatenation of both vector values then multiply them with parameter matrix. Pass this value to an activation function which is a nonlinear function like tanh (.). If the vectors of children are "n" dimensional then parent vector is also "n" dimensional. Repeat the above process for each level (Fig.7).

We can represent this whole processing with the help of binary tree structure. Where auto encoder is used at each node. Here we set $P = \frac{p}{||p||}$

$$P = \varphi^{(1)} (w^{(1)} [k_1; k_2] + b^{(1)}) \tag{6}$$

Where $[k_1; k_2] \in R\ 2^{n*1}$ and $w^{(1)} \in R^{n*2n}$ and $b^{(1)}$ is a bias which belongs to R^{n*1} and $\varphi^{(1)}$ is the element-wise activation function that is tanh(.).Here $k_1' = \frac{k'}{||k_1'||}$ and $k_2' = \frac{k'}{||k_2'||}$

$$[k_1' ; k_2'] = \varphi^{(2)} (\omega^{(2)} \pi + \beta^{(2)}) \tag{7}$$

Where k_1 and k_2 are the reconstructed children, $w^{(2)}$ is a parameter matrix for reconstruction, $b^{(2)}$ is a bias for reconstruction and $\varphi^{(2)}$ is element-wise activation function.

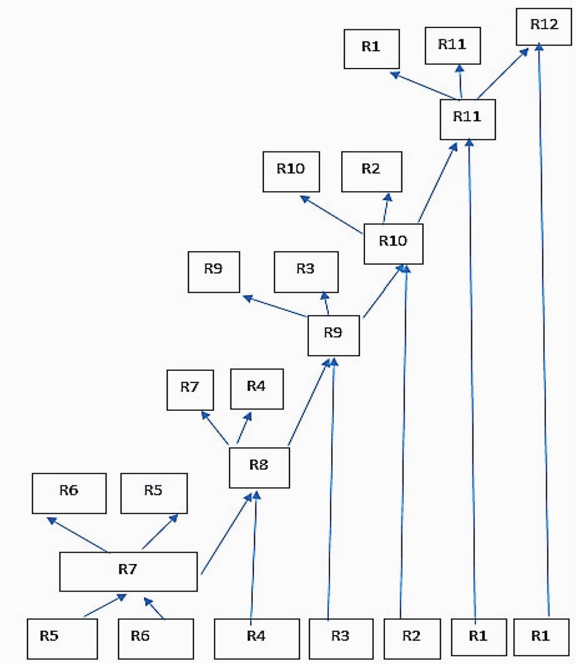


Fig. 7. Implementation of Reordering Rules

In the binary tree set is in the triplet form ($p \rightarrow k_1\ k_2$) where p is the parent vector and k_1 and k_2 are the children of the parent p.

In binary tree triplet's set's representation represented as ($v_1 \rightarrow u_1, u_2$), ($v_2 \rightarrow v_1, u_3$) and ($v_3 \rightarrow v_2, u_4$).

In recursive auto-encoder three steps are involved.

1. θ_{rec} = parameter matrix of recursive auto-encoder w_1 and w_2 and bias for both languages source and target.
2. θ_{reo} = parameter matrix w_1' and w_2' and the bias b^{o}.
3. θ_w = word embedding matrix for both languages source and target.

There are two types of error computing

1. Reconstruction error
2. Reordering error

Reconstruction error represents the vector space representation corresponding string.

For measuring the reconstruction error, we use the Euclidean distance between the input vector and reconstructed vector.

$$E_{rec}([k^1; k^2]; \theta) = \frac{1}{2} \| [k^1; k^2] - [k^{1'}; k^{2'}] \|^2 \qquad (8)$$

For calculating the reconstruction error, we use the greedy strategy.

Let us assume Erec ([u1; u2]; θ), Erec ([u2; u3]; θ), Erec ([u3; u4]; θ) are the sets of the sentence and Erec ([u1; u2]) the smallest error compare to all then the greedy strategy algorithm select it and replace u1 and u2 with their vector representation v1 generate by the recursive auto encoder then the strategy compute the Erec ([v1; u3]; θ), Erec ([u2; u3]; θ) and Erec ([u3; u4]; θ) and repeat these all above steps when we get the only one vector remain.

Giving the training set S = {t_i} where t_i ={ $o_i u_i^1 u_i^2$}.

For find average reconstruction error:

$$E_{rec,s}(S; \theta) = \frac{1}{N_s} \sum_i \sum_{p \in T_R^{\theta}(t_i, s)} E_{rec}([p.k_1, p.k_2]; \theta) \qquad (9)$$

Here $T_R^{\theta}(t_i, s)$ denotes the immediate nodes of source side binary tree and Ns is the number of occurring immediate nodes.

$p.k_n$ is the number of n^{th} child vector of parent p. + $E_{rec, t}(S; \theta)$ denotes the reconstruction error of target side.

$$E_{rec,s}(S; \theta) = E_{rec}(S; \theta) + E_{rec,t}(S; \theta) \qquad (10)$$

Reordering error represent the merging order with the help of classifier prediction.

Given training set t_i ={ $o_i u_i^1 u_i^2$}

$$E_k(t_i; \theta) = -\sum_o d_t(O) \log (p_\theta (O \mid C^1, C^2)) \qquad (11)$$

Here d_t is the probability of label. If O_i = straight then the value of label is [1, 0] and O_i = inverted then the value of label is [0, 1].

Where O ϵ {straight, inverted}, the reordering error is

$$E_{reo}(S; \theta) = \frac{1}{|S|} \sum_i E_c(t_i; \theta) \qquad (12)$$

The joint training function is

$$J = \gamma E_{rec}(S; \theta) + (1-\gamma) E_{reo}(S; \theta) + R(\theta) \qquad (13)$$

Where γ using for giving the preference between the reconstruction and recording error and

R (θ) is regularizer.

$$R(\theta) = \frac{\mu_w}{2} \|\theta_w\|^2 + \frac{\mu_{rec}}{2} \|\theta_{rec}\|^2 + \frac{\mu_{reo}}{2} \qquad (14)$$

We are using the greedy strategy for constructing the binary tree to represent each phrase level and these fixed binary tree's derivatives are calculated by backpropagation using structure.

RNN can be used with RAE in language modelling to predict more accurate sequence of words [14].

V. PERFORMANCE IMPROVEMENT USING GPUs

Deep learning application requires high computations because there exists large matrix multiplication, parallel processing and number of calculations during training phase.

Graphics processor unit (GPU) is very good option for parallel processing and fast computation as compare to the CPU. We are using NVIDIA GeoForce GTX TitanX to train word2vec for large corpus (3GB wiki data). It can also be used in training of recursive auto encode and recurrent neural network. GPU not only provides better energy efficiency but it also archives substantially higher performance over CPUs [1] [12].

VI. CONCLUSION

In the present time, machine translation is a very hot research topic in natural language processing area. Deep learning helps to train a translation system like a human brain. RNN, RAE provides better result in text processing as compare to other neural networks. Word alignment, reordering and language modeling can be performed with the help of a well-trained deep neural network. Word2vec generates the word-vectors that are used by recurrent auto-encoder in reconstruction task. RNN has the capability to implement reordering rules on sentences. GPU solves the problem of complex computation and leads the system towards good performance because it supports massive parallel computation.

VII. FUTURE WORK

Machine translation using deep learning is a good idea but it is very far from perfection. There exists lots of problems like lack of vocabulary, data sparseness, maintain history of vector values etc. A machine translation need very large corpus for it. Problem of gradient decent is also encounter when RNN is used, one solution is LSTM networks. Working with deep LSTM is better choice to build a more perfect translation system. Multiple GPUs can be used to accelerate training process. By implementing all these concepts, we will move towards an optimized machine translation system.

REFERENCES

[1] Daniel Schlegel, "Deep Machine Learning on Gpu", University of Heidelber-Ziti, 12 January 2015.

[2] Holger Schwenk, Yoshua Bengio," Learning Phrase Representations Using Rnn Encoder– Decoder for Statistical Machine Translation", Arxiv: V3 ,3 Sep 2014.

[3] Hugo Larochelle, Yoshua Bengio, Jerome Louradour, Pascal Lamblin, "Exploring Strategies for Training Deep Neural Networks", Journal of Machine Learning Research 1 2009 Submitted 12/07.

[4] Ilya Sutskever, Oriol Vinyals, Quoc V. Le, "Sequence to Sequence Learning with Neural Networks", Google.

[5] Jiajun Zhang And Chengqing Zong, "Deep Neural Network in Machine Translation", Institute of Automation, Chinese Academy of Sciences.

[6] Josep Crego, Jungi Kim, Guillaume Klein, Anable Rebollo, "Systran's Pure Neural Machine Translation System", Volume 1, 18 October 2016.

[7] Kyunghyun Cho, "From Sequence Modeling to Translation", Institut De Montréal Des Algorithmes D'apprentissage, Département D'informatique Et De Recherche Opérationnelle, Facult Des Arts Et Des Sciences, Université De Montréal

[8] Li Deng And Dong Yu, "Deep Learning: Methods and Applications", Microsoft Research One Microsoft Way Redmond, Wa 98052; Usa, Vol. 7, Nos. 3–4 (2013) 197–387.

[9] Martin Sundermeyer, Ralf Schluter, And Hermann Ney, "Lstm Neural Networks for Language Modelling", Huma Language Technology and Pattern Recognition, Computer Science Department, Rwth Aachen University, Aachen, Germany.

[10] Mohamed Amine Cheragui, "Theoretical Overview of Machine Translation", African University, Adrar, Algeria, Icwit 2012.

[11] Rico Sennrich, "Neural Machine Translation", Institute for Language, Cognition and Computation University of Edinburgh May 18 2016.

[12] Seulki Bae and Youngmin Yi, "Acceleration of Word2vec Using Gpus", School of Electrical and Computer Engineering, University of Seoul, Seoul, Republic of Korea, Springer International Publishing Ag 2016.

[13] Shahnawaz, R. B. Mishra, "A Neural Network Based Approach for English To Hindi Machine Translation", International Journal of Computer Applications Volume 53, September 2012.

[14] Shijie Liu, Nan Yang, Mu Li And Ming Zhou, "A Recursive Recurrent Neural Network for Statistical Machine Translation", Microsoft Research Asia, Sbeijing, China and University of Science and Technology of China, Hefei, China, Baltimore, Maryland, Usa, June 23-25-2014.

[15] Soheil Baharampour, Naven Ramkirishan, Lukas Schott, Mohak Shah, "Comparative Study Of Deep Learning Software Frameworks", Research and Technology Center, Robert Bosch Llc.

[16] Wei He, Zhongjun He, Huawu, And Haifengwang, "Improved Neural Machine Translation With Smt Features", Proceedings of The Thirtieth Aaai Conference On Artificial Intelligence (Aaai-16) Volume 3, 30 March 2016.

[17] Younghui Wu, Mike Schuster, Zhifeng Chen, "Google's Neural Machine Translation System: Bridging the Gap Between Human and Machine Translation", Volume 2, 8 October 2016.

[18] http://www.deeplearningbook.org/.

[19] http://haifux.org/lectures/267/introduction-to-gpus.pdf

[20] https://arxiv.org/pdf/1411.2738.pdf

[21] http://nlp.cs.tamu.edu/resources/wordvectors.ppt

[22] http://www.minerazzi.com/tutorials/cosine-similarity-tutorial.pdf