Neural Machine Translation for English to Hindi

Sandeep Saini
Department of Electronics and Communication Engineering
The LNM Institute of Information Technology
Jaipur, India
sandeep.saini@lnmiit.ac.in

Vineet Sahula
Department of Electronics and Communication Engineering
Malaviya National Institute of Technology
Jaipur, India
sahula@ieee.org

***Abstract*—Language translation is one task in which machine is definitely lagging behind the cognitive powers of human beings. Statistical Machine Translation is one of the conventional ways of solving the problem of machine translation. This method requires huge data sets and performs well on similar grammar structured language pairs. In recent years, Neural Machine Translation (NMT) has emerged as an alternate way of addressing the same issue. In this paper, we explore different configurations for setting up a Neural Machine Translation System for Indian language Hindi. We have experimented with eight different architecture combinations of NMT for English to Hindi and compared our results with conventional machine translation techniques. We have also observed in this work that NMT requires very less amount of data size for training and thus exhibits satisfactory translation for few thousands of training sentences as well.**

Keywords*—*Cognitive linguistics, Neural Machine Translation, Long and Short Term Memory, Indian Languages

I. Introduction

Machine translation was one of the initial tasks taken by the computer scientists and the research in this field is going on for last 50 years. During these years, it has been a remarkable progress that linguists and computer engineers have worked together to achieve the current status of machine translation. Machine Translation task was initially handled with dictionary matching techniques and slowly upgraded to rule-based approaches. During last two decades, most of the machine translation systems were based on statistical machine translation approach. In these systems, [1], [2] and [3], the basic units of translation process are phrases and sentences. These phrases can be composed of one or more words. Most of the conventional translation systems are based on Bayesian inferencing to predict the estimate translation probabilities for pairs of phrases. In these pairs, one phrase belongs to the source language and the other from the target language. Since the probability of these phrases is very low, thus pairing and predicting the correct pair is very difficult in these systems. To improve the probability of a certain pair of phrases, increasing the size of the dataset was one of the most feasible solutions. With the limitations of conventional Machine Translation Systems [4] and dependence on huge datasets, there is a demand to search for alternate methods for machine translation.

In recent years, Google has also shifted its translation research focus towards Neural Machine Translation (NMT). Sutskever, et al. [5] proposed a sequence to sequence learning mechanism using long and short-term (LSTM) memory models. This neural network-based machine translation system had eight layers of encoder and eight layers of the decoder. The core idea of NMT is to use deep learning and representation learning. NMT models require only a fraction of the memory needed as compared to the traditional statistical machine translation (SMT) models. Furthermore, unlike conventional translation systems, all parts of the neural translation model are trained jointly (end-to-end) to maximize the translation performance [5] and [6]. In an NMT system, a bidirectional recurrent neural network (RNN), known as an encoder, is used by the neural network to encode a source sentence for a second RNN, known as a decoder, that is used to predict words in the target language. This encoder-decoder architecture can be designed with multiple layers to increase the efficiency of the system.

Normally neural machine translation tends to require a lot of computing power which means that it is normally a great technique if we have enough time or computing powers. The other issue with older NMT was inconsistency in handling rare words. Since these inputs were sparsely available in the network, the learning and inferencing were not efficient. By using LSTM models and having eight layers of encoder and decoder, this system removes these errors to a large extent. The third major issue with NMT was that the system used to forget the words after a long. This issue is also resolved in 8 layer approach. After 2014, this work from Sutskever et. Al. have inspired many researchers and NMT is developing as a good alternative to conventional machine translation techniques.

Google has deployed GNMT on an eight-layer encoder-decoder architecture. This system requires huge GPU computations for training the neural network. In this work, we explore a simplified and shallow network that can be trained using a regular GPU as well. We have explored different architectures of the shallow network and shown satisfactory results for the Indian language.

II. Neural Machine Translation

Neural Machine Translation (NMT) is a machine translation system that uses an artificial neural network to increase fluency and accuracy the process of machine translation. NMT is based on a simple encoder-decoder based network. The type of neural networks used in NMT is Recurrent Neural Networks (RNN) [7]. The reason for selecting RNN for the task is the basic architecture of the RNN. RNN involves cyclic structure and which enables the learning of repeated sequences much easier than the other networks. RNN can be unrolled to store the sentences as a sequence in both sources as well as target languages. A typical structure for RNN is described in Fig. 1. This explains that how a single layer can be unrolled into

multiple layers and information of the previous time period can be stored in the single cell as well. RNNs can easily map sequences to sequences when the alignment between inputs and outputs is known ahead of time.

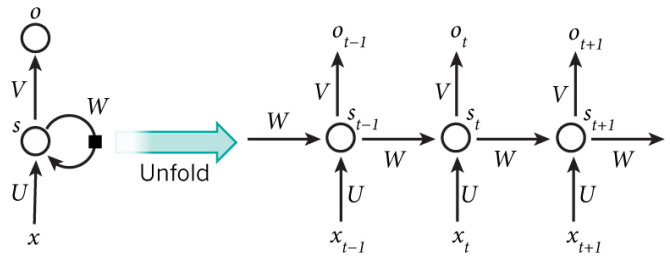

Fig. 1. Typical structure of an RNN

Let X and Y be the source and target sentence pairs respectively. The encoder RNN converts the source sentence $x_1, x_2, x_3, ...x_n$ into different vectors of fixed dimensions. The decoder will provide one word at a time as its output, using conditional probability

$$P(Y|X) = P(Y|X_1, X_2, X_3, ...X_m) \tag{1}$$

Here $X_1, X_2.....X_M$ are the fixed size vectors encoded by the encoder. Using chain rule the above equation is converted to the equation below where while decoding, next word is predicted using symbols that are predicted till now and source sentence vectors. The above expression then becomes

$$P(Y|X) = P(y_i|y_0, y_1, y_2, ..., y_{i1}; X_1, X_2, X_3, ...X_m) \tag{2}$$

Each term in the distribution is represented by a softmax function over all the words in the vocabulary.

Although RNNs work very well, in theory, there are problems while training them with long sentences because RNNs have a "Long Term Dependency Problem." The reason is that in RNN the final decision at time t is computed as

$$\frac{\partial h_t}{\partial h_0} = \frac{\partial h_t}{\partial h_{t-1}} * \frac{\partial h_{t-1}}{\partial h_{t-2}} * ... \frac{\partial h_2}{\partial h_1} * \frac{\partial h_1}{\partial h_0} \tag{3}$$

Thus because of multiplicative effect, the output in longer sentences is very low and results in inaccuracy. This can be explained by a simple example. If the given task is to predict the next word in a sentence using a language model, then in a sentence like "Potholes are located on the _____ , **road** is the obvious choice to fill in the blank, and we do not need any further context as the gap between the relevant information, and its place is small. However, in sentences like I am a cricket player and I bat well at the _____ position. it is not as straightforward because from the recent information we can only deduce that missing word should be a position in the batting order. However, there can be multiple choices (e.g., opening, middle order and lower order). This problem is felt when the gap between the relevant information and the place that it is needed is not small, and it is entirely possible to have gaps much bigger than in this example. In practice, it is difficult for RNNs to learn these dependencies. Since the typical sentences in any language have such complex context-dependent cases, then RNN should not be used for encoder and decoder design. To overcome the shortcomings of the RNNs, we use Long and Short Term Memory (LSTM) models for encoding and decoding.

A. Long and Short Term Memory (LSTM) model

Long and Short Term Memory [8] is a variation of RNN and are known to learn problems with long-range temporal dependencies, so RNNs are sometimes replaced with LSTMs in MT networks. LSTMs also have this chain-like structure, but the structure of the repeating module is different from RNN. In place of a single neural network layer, there are four layers in a module. These layers interact within the same modules as well as with other modules for learning. A typical structure of LSTM module is shown in Fig 2.

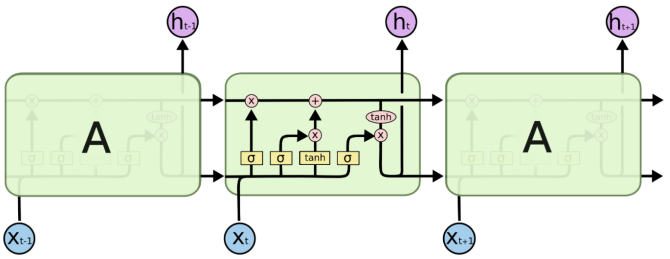

Fig. 2. The repeating module in an LSTM contains four interacting layers

In this module, there are four gates for four different operations in the learning process. The first gate if "forget gate" (f_t. This helps in deciding, which part of the previous learning should be forgotten in this layer. The next is a sigmoid layer called the input gate layer (i_t) decides which values the system will update. The third gate is a tanh layer that creates a vector of new candidate values, $\widetilde{C_t}$, that could be added to the state in the same module. Finally, the output is decided by the fourth layer. This is also a tanh function which generates the state for next modules. The corresponding equations for all these functions are as follows:

$$f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f) \tag{4}$$

$$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i) \tag{5}$$

$$\widetilde{C_t} = tanh(W_C * [h_{t-1}, x_t] + b_C) \tag{6}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \tag{7}$$

$$o_t = \sigma(W_o * [h_{t-1}, x_t] + b_o) \tag{8}$$

$$h_t = o_t * tanh(C_t) \tag{9}$$

In Fig. 3, the model reads an input sentence ABC and produces WXYZ as the output sentence. The model stops making predictions after generating the end-of-sentence (EOS) token as the output.

III. Setting up the Neural Machine Translation System

A. Encoder and Decoder

In LSTM based NMT, we use a bidirectional encoder [5]. This encoder is based on the concept that the output at any time instant may not only depend on past data but also on future data. Using this idea, the LSTM is tweaked to connect two hidden layers of opposite directions to the same output. This tweaked version of LSTM is called a Bidirectional LSTM (Bi-LSTM) [9]. Bi-LSTMs were introduced to increase the amount of input information available to the network. Unlike LSTMs, Bi-LSTMs have access to the future input from the current state without the need for time delays. Fig. 4 shows the architecture for the bidirectional Encoder used in our NMT system. The encoder presented in this figure is a single layer encoder. In GNMT, eight layer encoder and decoder clocks are used to process the information. For better efficiency in the learning process, multiple layers of LSTMs are preferred in encoder as well as decoder designs.

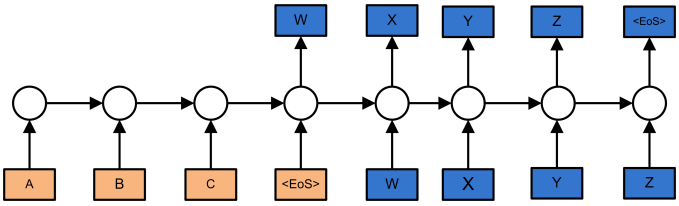

Fig. 3. Sentence modeling in LSTM network

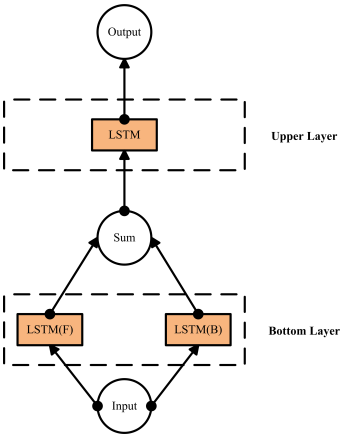

Fig. 4. Bidirectional Encoder design using Bi-LSTM

The decoder is designed to decode the vectors back to target language words. We have experimented with multi-layer decoders in the system. A typical two-layer decoder is shown in Fig. 5.

B. Attention in the model

Attention layer is the bridging layer between encoder and decoder of an NMT system. There are two kinds of attention models; global and local. The idea of a global attention model is to consider all the hidden states of the encoder when deriving the context vector c_t. In global attention model, a_t, which is a variable-length alignment vector with a size equals to the number of time steps on the source side, is derived by

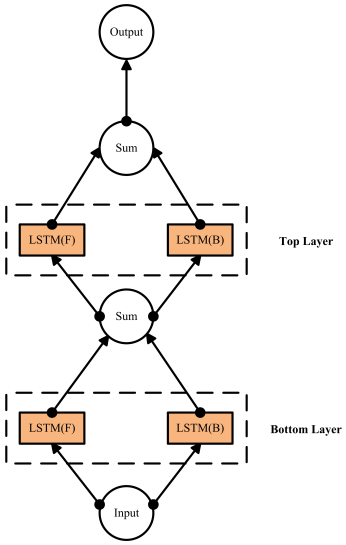

Fig. 5. 2 layer decoder architecture

comparing the current target hidden state h_t with each source hidden state h_s. The concept of modeling the language is different in local attention model. In local attention model, the model first predicts a single aligned position p_t for the current target word. With the help of a window, which is centered around the source position, p_t computes a context vector c_t.

In our system, we have used local attention model. A block diagram showing the functionality of attention layer is shown in Fig 6.

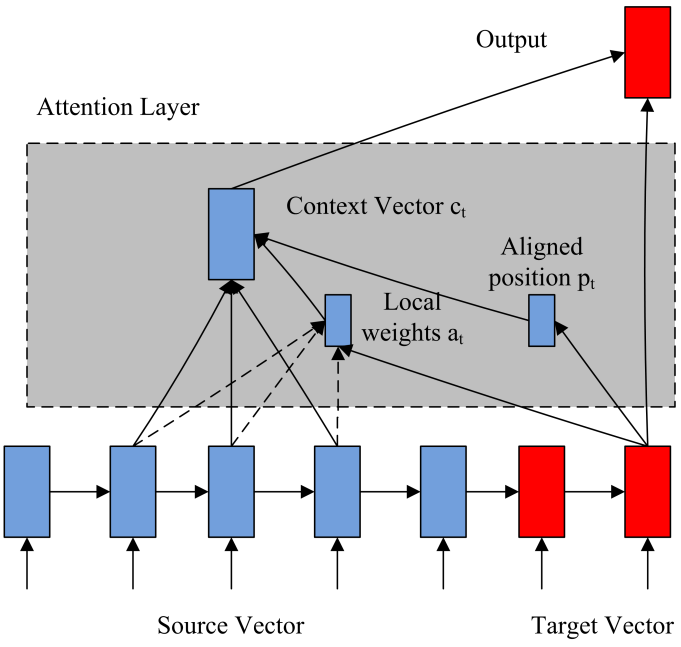

Fig. 6. Local attention model.

C. Residual connections and Bridges

The success of a neural network depends on the depth of the network. However, as the depth of network increases, it becomes more and more difficult to train, due to vanishing and

exploding gradients [10]. This problem has been addressed in the past using the idea of modeling differences between an intermediate layers output and the targets. These are called residual connections. With residual connections, the input of a layer is added elementwise to the output before feeding to the next layer. In Fig 7, the output of LSTM1 is added to input and sent as an input to LSTM2. Residual connections are known to greatly improve the gradient flow in the backward pass, which allows us to train very deep networks.

An additional layer is needed in between encoder and decoder layers. Fig 8 shows the complete system consisting of Encoder, decoder, residual connections and bridge. Figure 9 shows the graphical representation of how sentences are converted in vectors and associated with vectors in the target language.

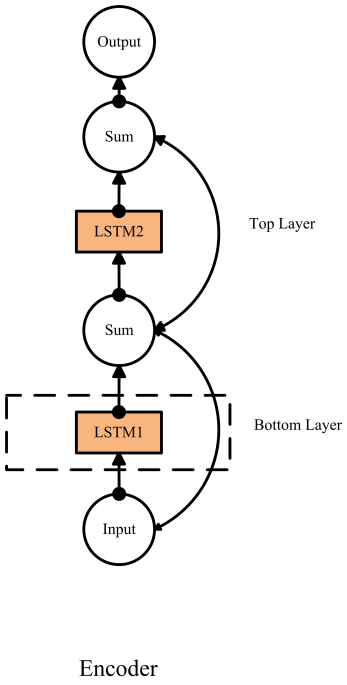

Fig. 7. Residual connections inside the Encoder

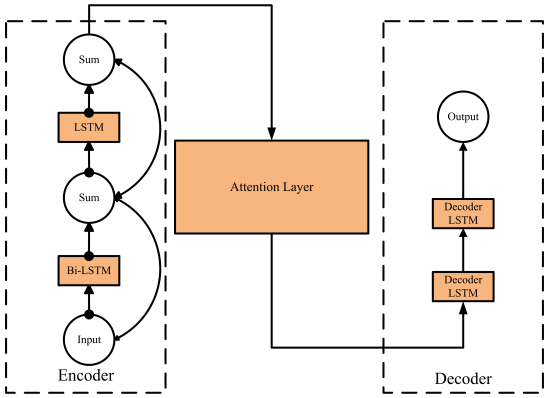

Fig. 8. Complete system block diagram for NMT

IV. Results and Discussion

A. Dataset

The initial requirement for setting up a machine translation system is the availability of parallel corpus for source and target languages. Hindi is not as resourceful language as its European counterparts in terms of availability of large datasets. Premier institutes in India and abroad have been working from past 2 decades on the development of parallel corpus for Indian languages. We have considered the following three different datasets for the experiments.

1) English-Hindi parallel corpus from Institute for Language, Cognition, and Computation, the University of Edinburgh [11]
2) Institute of Formal and Applied Linguistics (UFAL) at the Computer Science School, Faculty of Mathematics and Physics, Charles University in Prague, Czech Republic [12]
3) Center for Indian Language Technology (CFILT), IIT Bombay [13].

Datasets from ILCC, University of Edinburgh, contains translated sentences from Wikipedia. IIT Bombay and UFAL datasets contain the data from multiple disciplines. All these datasets are exhaustive with an abundant variety of words. Table I provides the information regarding the number of words and sentences in each dataset.

TABLE I. Details of ILCC and UFAL Hindi datasets

	ILCC Dataset	**UFAL Dataset**	**CFILT Dataset**
No of sentences	41,396	237,885	1,492,827
No of Words	245,675	1,048,297	20,601,012
No. Of Unique Words	8,342	21,673	250,619

B. Experimental setup

This NMT system is implemented by taking core algorithm from the tutorials from Peter Neubig [14] and [1]. TensorFlow[2] and Theano[3] are the platforms used in the system design. We have set up the system on a Nvidia GPU, which is having NVIDIA Quadro K4200 graphics card. this GPU has 24 stacks and a total number of CUDA cores 1344.

C. Training details

Once the dataset is preprocessed, the source and target files are fed into the encoder layer to prepare the vectors from the sentences. We have used Stochastic gradient descent (SGD) [15], an algorithm for training. We have worked on two different layer sizes. Two and four layer networks are trained for different combinations of encoder and decoder architectures. We replace LSTM with Bi-LSTM and also experimented with Deep Bi-LSTM. We also add residual connections and attention mechanism. We also add a dense layer which acts as a bridge between encoder and decoder and compared its performance with other methods.

[1] https://google.github.io/seq2seq/nmt/
[2] https://www.tensorflow.org/install/
[3] http://deeplearning.net/software/theano/install.html

TABLE II. TRAINING DETAILS FOR 3 DATA SETS

Dataset	Total Sentences	Training	Validation	Testing
ILCC	41,396	28,000	6,000	6,000
UFAL	237,885	70,000	15,000	15,000
CFILT	1,492,827	140,000	30,000	30,000

TABLE III. TRAINING TIME FOR DIFFERENT CONFIGURATIONS OF ENCODER AND DECODERS FOR NMT

		Training Time (hh:mm:ss)			
Data set	**No. of sentences**	**2 Layer LSTM + SGD**	**4 Layer LSTM + SGD**	**2 Layer (Bi-dir) LSTM +SGD + Res**	**4 layers (Bi-dir) LSTM +SGD+ Res**
ILCC	28,000	02:58:35	6:34:54	3:28:34	7:45:56
UFAL	70,000	07:34:28	13:46:25	8:31:24	15:23:41
CFILT	140,000	16:38:24	28:38:12	15:43:23	32:25:16

Since we have used GPU, training time for the neural network for different datasets for different architectures was in few hours only. For the experiment, we have used a different number of sentences for each data set. Details about a number of sentences used in training and testing for each data set is described in Table II. Training time for each dataset for a selected number of sentences is shown in table III. Every dataset is trained for 10 epochs.

D. BLEU Score

We have evaluated our system using BLEU score [16]. In each configuration, BLEU score of the translation scores are different and table IV shows the BLEU score for each different configuration.

E. Discussions

The results obtained from NMT based English-Hindi machine translation is comparable with conventional statistical or phrase-based machine translation systems. One of the earliest SMT based system Anusaaraka [17] is lacking the capability to handle complex sentences and doesn't perform at par with the latest MT systems. In our work, we have also inspired the task of translation from human cognitive abilities to translate the language. This system does not outperform GNMT (with a BLEU score of 38.20 for En-Fr), but it is showing many comparable results, when compared to Anusaaraka (21.18), AnglaMT (22.21) and Anglabharati (20.66) [18].

V. CONCLUSION

Statistical Phrase-based Machine translation systems have been facing the problem of accuracy and requirement of large data sets for a long time, and in this work, we have investigated the possibility of using a shallow RNN and LSTM based Neural Machine translator for solving the issue of Machine Translation. We have used quite a small amount of dataset and less number of layers for our experiment. The results show that NMT can provide much better results for the larger dataset and have a large number of layers in encoder and decoder. Compared to contemporary SMT and PBMT systems, NMT based MT performs much better. Future work would involve in fine-tuning the training of long and rare sentences using smaller data sets. We would like to explore NMT for Indian language pairs as well. Since the grammar structure for many of the Indian languages is similar to each other, we expect the higher order of BLEU scores in future.

TABLE IV. BLEU SCORE CALCULATED FOR 4 DIFFERENT CONFIGURATIONS OF NMT SYSTEM

	BLEU Score for En-Hi		
Configuration	**ILCC**	**UFAL**	**CFILT**
2 Layer LSTM + SGD	12.512	14.237	16.854
4 Layer LSTM + SGD	13.534	16.895	17.124
2 Layer (Bi-dir) LSTM +SGD	12.854	15.785	18.100
4 layers (Bi-dir) LSTM +SGD+ Res	13.863	17.987	18.215

REFERENCES

[1] A. Lavie, S. Vogel, L. Levin, E. Peterson, K. Probst, A. F. Llitjós, R. Reynolds, J. Carbonell, and R. Cohen, "Experiments with a hindi-to-english transfer-based mt system under a miserly data scenario," vol. 2, no. 2, pp. 143–163, Jun. 2003. [Online]. Available: http://doi.acm.org/10.1145/974740.974747

[2] S. Saini, U. Sehgal, and V. Sahula, "Relative clause based text simplification for improved english to hindi translation," in *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Aug 2015, pp. 1479–1484.

[3] S. Saini and V. Sahula, "A survey of machine translation techniques and systems for indian languages," in *2015 IEEE International Conference on Computational Intelligence Communication Technology*, Feb 2015, pp. 676–681.

[4] S. Chand, "Empirical survey of machine translation tools," in *2016 Second International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, Sept 2016, pp. 181–185.

[5] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *CoRR*, vol. abs/1409.3215, 2014. [Online]. Available: http://arxiv.org/abs/1409.3215

[6] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[7] L. Medsker and L. Jain, "Recurrent neural networks," *Design and Applications*, vol. 5, 2001.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[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] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International Conference on Machine Learning*, 2013, pp. 1310–1318.

[11] "Institute for language, cognition and computation, university of edinburgh, indic multi-parallel corpus, http://homepages.inf.ed.ac.uk/miles/babel.html," Tech. Rep.

[12] O. Bojar, V. Diatka, P. Rychlỳ, P. Stranák, V. Suchomel, A. Tamchyna, and D. Zeman, "Hindencorp-hindi-english and hindi-only corpus for machine translation." in *LREC*, 2014, pp. 3550–3555.

[13] "Resource centre for indian language technology solutions(cfilt) i.-b. h. corpus, http://www.cfilt.iitb.ac.in/downloads.htm," Tech. Rep.

[14] G. Neubig, "Neural machine translation and sequence-to-sequence models: A tutorial," *arXiv preprint arXiv:1703.01619*, 2017.

[15] D. Needell, R. Ward, and N. Srebro, "Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm," in *Advances in Neural Information Processing Systems*, 2014, pp. 1017–1025.

[16] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: A method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ser. ACL '02. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, pp. 311–318. [Online]. Available: http://dx.doi.org/10.3115/1073083.1073135

[17] A. Bharati, V. Chaitanya, A. P. Kulkarni, and R. Sangal, "Anusaaraka: Machine translation in stages," *CoRR*, vol. cs.CL/0306130, 2003. [Online]. Available: http://arxiv.org/abs/cs.CL/0306130

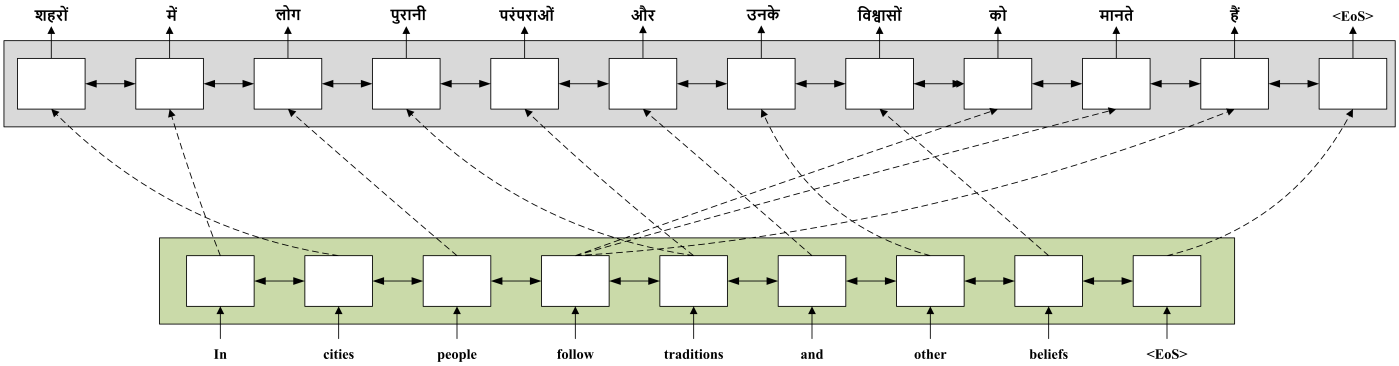

Fig. 9. Graphical representation of vector relations between source and target sentences.

[18] K. Sachdeva, R. Srivastava, S. Jain, and D. M. Sharma, "Hindi to english machine translation: Using effective selection in multi-model smt." in *LREC*, 2014, pp. 1807–1811.