Machine Translation

(Hindi to English Translation)

NAME	ROLL NO.	EMAIL
Shivam Shukla	MT2017108	Shivam.shukla@iiitb.org
Shubham Bansal	MT2017115	Shubham.bansal@iiitb.org
Hatim Bohra	MT2017138	Hatim.bohra@iiitb.org

1. **Problem Statement**: our main task is to convert Hindi sentence to English sentence

   Input :क्या यह किताब तुम्हारी है?

   Output:Is this book yours?

   Input :मेरी बात ने उसके अभिमान पर चोट पहुँचाई।

   Output: What I said hurt his pride.

2. **Dataset description**: we used a Hindi to English dataset from the given website DATASET LINK It has 2867 sentences and covers almost all types of variability in the sentences which are used on the daily basis.

3. **Architecture/algorithm details:**

   a)

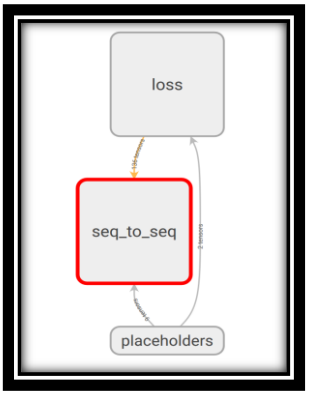

Fig- Abstract view of the model

b)

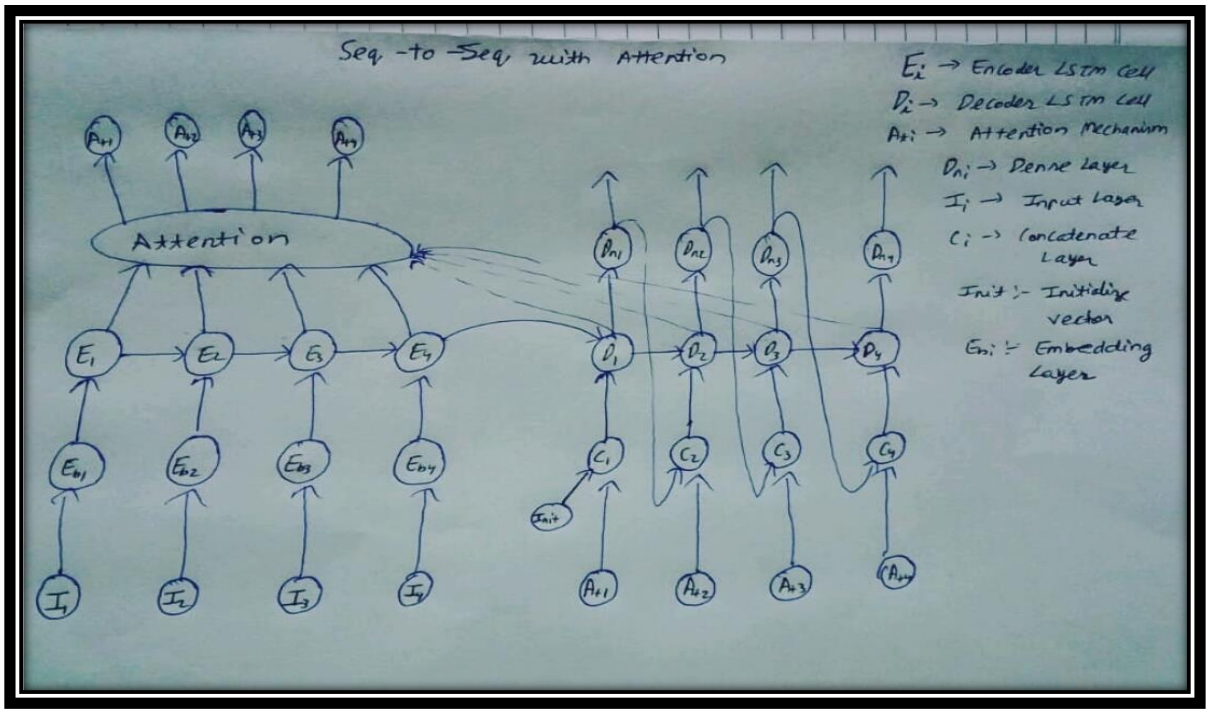

Fig- Sequence to sequence architecture

c)

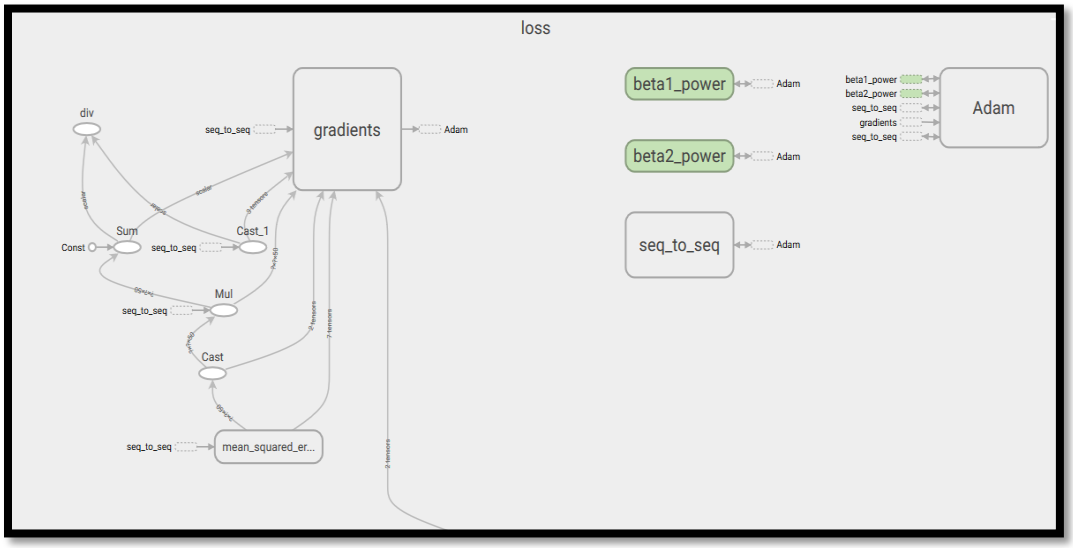

Fig- Mean square loss and optimizer

4. Implementation/training/test strategy

Preprocessing:

1) We processed the dataset so as to remove the punctuations present there using regular expression library.

2) Spell Corrections for English words: - Using autocorrect library in order to make corrections, if any Misspelled words are present.

3) Convert all upper-case letter to lower case counterparts in order to make our dataset homogeneous.

4) Adding to previous work, we prepared vec dictionary corresponding to every word and further assigning 50-dimension glove vector embedding which is trained on Wikipedia datasets (which houses 40K words embedding) by Stanford.

5) There are words (may be few or more) which are not present in the embedding file. This steps basically deals with finding those words.

6) The words which were found absent in the previous step are very few mostly belong to noun category. So we simply replaced them by a letter 'a'(as flexible as our wish).

7) We used Integer encoding for Hindi words by using **sklearn LabelEncoder**

8) We have assumed the maximum length of English and Hindi sentences to be of 30 words which will not be less than the length of any sentence present in our dataset.

9) Then we padded all those English and Hindi sentences which were falling short from maximum length of 30 words, with '0' and #' respectively.

10) Preparing arrays for English sentences length and Hindi sentences length Now we prepared array of English and Hindi sentences.

11) At the end, we computed MSL for loss computation.

Implementation: -

Making Computational graph in TensorFlow

Following layers are used in TensorFlow

1)Placeholders:

i) Hindi sentences representation as Inputs in TensorFlow

graph shape: - (batch_size,maxhindiword) ditype:- int

ii) English sentences representation as a Labels in tensorflow graph

shape:- (batch_size,maxengword,emb_size) dtype :-float

iii) src_seq_len:- shape:- (batch_size,) dtype:- int(len. of hin sentence for eachdata)

iv) trg_seq_len:- shape (batch_size,) dtype:- int (length of eng sentence)

v) msl:- shape(batch_size,) dtype: float

2) *Embedding Layer:* converting hindi vocabulary to embedding size 50 by *tf.embedding_lookup* method

3) Seq_to_Seq:

For Encoder and Decoder which uses lstm cell has hidden vector size of 30.

Encoder takes inputs from embedding layer which is size 50 and from previous hidden and cell state of encoder and output new hidden and cell vectors:-

Decoder takes inputs from previous dense layer and from attention layer concatenate it and form vector of (50+30) dimension inputs size where 50 is target embedding size and 30 is attention vector and from previous hidden and cell state of encoder and output new hidden and cell vectors.

4) *Attention mechanism*:

This layer process for every time step at decoder and it takes inputs from encoder network for all sequences and from decoder network for particular time step as a context vector and applying following operation which given in equation below for computation alpha and it use as a weights and compute weighted sum of all hidden vector in encoder sequence for every time step this implies hidden vector in encoder network which has high alpha has high contribution means high attention.

We used following equation for attention:

$$s(q,r) = \mathbf{u}^T f\left(\mathbf{v}_q^T \mathbf{M}^{[1:a]} \mathbf{v}_r + \mathbf{V} \begin{bmatrix} \mathbf{v}_q \\ \mathbf{v}_r \end{bmatrix} + \mathbf{b} \right) \quad (4)$$

where f is a standard nonlinearity applied element-wise, $\mathbf{V} \in \mathcal{R}^{a \times 2n_s}$, $\mathbf{b} \in \mathcal{R}^a$, $\mathbf{u} \in \mathcal{R}^a$, $\mathbf{M}^{[1:a]} \in \mathcal{R}^{n_s \times n_s \times a}$ is a tensor and the bilinear tensor product $\mathbf{V}_q^T \mathbf{M}^{[1:a]} \mathbf{v}_r$ results in a vector $h \in \mathcal{R}^a$, where each entry is computed by one slice $i = 1, \ldots, a$ of the tensor $h_i = \mathbf{V}_q^T \mathbf{M}^i \mathbf{v}_r$.

Where **vq** represent hidden vectors in encoder network and **vr** represent a hidden vector at a particular time t of decoder network and **ns** is hidden_vector size

5) *Dense Layer* :- which takes inputs from decoder outputs and map 30 dim to 50 dim

6) *Mean Square Loss*:- It is a root mean square distance between dense layer output and labels output which computed distances between two vectors lower value less distance means less loss ,Optimizer :- Adam optimizer

***Evaluation criteria and Results**:*

For evaluation we find Euclidean distance between predicted embedding vector and actual embedding vector.

Experimentation and Results:

Loss start from 540 which is mean loss per epoch and keep decreasing to 330 and then remains constant.

Final prediction does not show valid sentences.

We computed gradients of mean square loss with respect to word embeddings and it shows that gradient value has been propagated to input layer.

We also exponentiate mean square loss for boosting gradients flow (as Prof. Raghavan advised) but it resulted in NAN value.

This might be the case that our dataset is not enough so as to capture all the variability or it may also be the case that the number of parameters are too much to learn from our existing dataset(which is quite small when compared to no of parameters involved in our model). That's why model does not reduces loss to further extent.