

# Machine Translation-Hindi to English

by  
Sravya K  
Aishwarya MG  
Arjun H

## Abstract

Hindi to English machine translation using neural machine translation is a supervised learning technique. The NMT algorithm is trained to learn to translate to English from Hindi. For the base model we have considered a recurrent neural network with GRU. A few improvements like Attention, GloVe embedding and Byte pair encoding techniques have been implemented to improve the Bilingual Evaluation Understudy (BLEU) score (which is our evaluation metric). This paper will describe the above-mentioned approaches and their results in comparison the baseline model.

## 1 Introduction

Our task is to translate Hindi texts to English. Machine translation using a neural network helps to translate a huge amount of data in a short period of time with better accuracy. It also helps to reduce the human labor and also eliminates the extra cost needed for human translators. Content can be shared among people all around the world. There are many cases that show how productive these translations can be. Google translate API is one of the the most famous machine translations system that translates thousands of languages in day to day life. IBM Watson is another example of a machine translation device. A lot of social networking sites like Facebook, Skype, Google talk, Messenger allow their users to speak in multiple languages in order to communicate with each other.

Many of our appliances like mobile phones, pocket PCs have machine translation applications

installed in them allowing networking between partners who speak different languages. Several other industries like government, health care, finance, Legal, Software and technology take advantage of machine translation. It really plays an important role in industries like health care where it is necessary to understand a person is saying to prevent hazardous situations. It helps patients to communicate better. Real time translation applications allow the text-to-text, text-to-speech, speech-to-text, speech-to-speech and Image-to -text translations. Our goal was to replicate these systems with better productivity.

Our project is helpful because not everyone speaks English. English is a global language and translation from any language to English reduces the barrier and helps people to access more information. Also, another application can be the way non-Hindi speakers can get access to the documents that are written in Hindi. The input in our project can be texts, sentences or words in Hindi from any source like books, newspapers, news channels, articles etc. The output should be English words or sentences which have the exact meaning of the input.

## 2 Electronically-available Resources

The related works we referred are Sequence to Sequence Learning with Neural Networks by Ilya Sutskever, Oriol Vinyals, Quoc V. Le (2014), Neural Machine Translation by jointly learning to align and Translate by Dzmitry Bahdanau Jacobs University Bremen, Germany Kyung, Hyun Cho, Yoshua Bengio and Effective Approaches to Attention-based Neural Machine Translation Minh-Thang Luong Hieu Pham Christopher D. Manning

The first paper marked the beginning of the use of Neural network using sequence to sequence learning. They wanted to show that a large deep Long Short-Term Memory with a limited vocabulary can outperform a standard SMT-based system whose vocabulary is unlimited on a large-scale MT task and is able to learn linguistic rules on its own from statistical models.

The second paper which we referred Neural machine translation aims at building a single neural network that can be jointly tuned to maximize the translation performance, unlike the traditional statistical machine translation. The models proposed recently for neural machine translation often belong to a family of encoder-decoders. In this paper, they wanted to prove that the use of a fixed-length vector is a bottleneck in improving the performance of the basic encoder-decoder architecture, and proposed to extend this by allowing a model to automatically (soft-)search for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly.

The third paper explores two effective mechanisms in which Neural Machine Translations with attention is being implemented. Firstly, a global approach in which all the hidden states of encoder are used to predict a target word. The second approach is a "Local Attention" approach in which we use only a subset of all source words to predict target word. This paper proves that this method improves BLEU score by 1 point that the existing current attention based NMT systems.

### 3 Data

Our data set consists of tab-delimited bilingual sentence pairs consisting of Hindi and English sentences. Input is Hindi sentences and output is English sentences. The data set has been downloaded from the site: [www.manythings.org/anki](http://www.manythings.org/anki) and [tatoeba.org](http://tatoeba.org). Many of the sentences came from the Tanaka Corpus which was imported into the Tatoeba Corpus. Labels for this are also sentences. These are human translated sentences. Each Hindi sentence has an equivalent English sentence in the dataset. The dataset has

one relevant English sentence for every Hindi sentence in the dataset. The dataset is quite small. It has 2785 sentences in both English and Hindi. We are splitting the data set into train and test in 80 to 20 ratios.

### 4 Methodology

In our Baseline model, we load the training Hindi and English data into a data frame, convert the English sentences to lower case. Then we tokenize the sentences and store the unique words. In the next step, we convert these tokenized texts into a vector(sequences) and then pad each sentence to have equal length. Our training set consists of 3000 units. We find the maximum length input vector among Hindi and English languages and pad our vectors to achieve the fixed-length vectors.

We have one input layer that also uses GRU and gets the output of GRU for each time step to pass to the next layer in the model. This method is called Teacher forcing, this model has one hidden layer which uses tanh as its activation function and then we have an output layer that outputs the softmax of the layer so that we can predict which word index has higher probability.

Once the model is trained, we test our model on the testing set with Hindi as input and we finally generate the BLEU score of the candidate sentences against the reference sentences.

As our first improvement, we implemented Attention. Attention is a mechanism introduced by Bahdanau that allows the model to learn to align and translate. In this mechanism, specific parts of input sequences are focused to obtain output sequences to get better results. Attention gives better results for inputs sequences longer than 30 characters. Using Attention, the alignment issues that can be observed in the output statements can be fixed.

To implement attention, we have to calculate the attention weights using encoder output and hidden state. A softmax is applied on the top, which when multiplied with the encoder output gives us context vector that tells us which context of the input given to the decoder we should focus on in the current unit of neural network

GloVe, a pre-trained word embedding is our second improvement. GloVe is an unsupervised learning algorithm in which training is performed on aggregated global word-word co-occurrence statistics which also preserves semantic analogy. GloVe reduces the number of steps needed for the output to converge as it is a pre-trained network.

In GloVe, as a part of first step, pre-trained word vectors are loaded. And then size of the embedding matrix is determined based on the vocabulary size. Any word that is not found in the embedding index will all be zero.

Our final improvement of the project is the usage of Byte-pair Encoding for sub word embeddings that is trained on Wikipedia corpus. word embeddings like word2vec or GloVe replace the out of vocabulary words with <unk> token and train a generic embedding for such unknown words. Using BPEmb, we reconstruct the meaning of the word from its parts(suffix, prefix). Internally this uses Sentencepiece to split the unknown words into known sub words. In this, we are allowed to pick the vocabulary size that determines the length of sub word.

In this approach, we encode the input sentences using bpemb model and then convert it into vector format. We send this sub word embedding vector as an input to the encoder. The output of the encoder is given to the decoder to generate the sub word embeddings for the English, which when joined produces our final output

## 5 Experiments

S.no	Model	BLEU Scores
1)	Baseline	BLEU-1: 0.200887  BLEU-2: 0.056415  BLEU-3: 0.037152  BLEU-4: 0.015138
2)	Baseline+Attention	BLEU-1: 0.285778

		BLEU-2: 0.148456  BLEU-3: 0.128605  BLEU-4: 0.078190
3)	Baseline+GloVe	BLEU-1: 0.309200  BLEU-2: 0.158939  BLEU-3: 0.139529  BLEU-4: 0.088947
4)	Baseline+Bpe	BLEU-1: 0.236064  BLEU-2: 0.115474  BLEU-3: 0.100143  BLEU-4: 0.057748
5)	Baseline+Attention +GloVe	BLEU-1: 0.472071  BLEU-2: 0.378231  BLEU-3: 0.364372  BLEU-4: 0.283516
6)	Baseline+Bpe +GloVe	BLEU-1: 0.274872  BLEU-2: 0.150744

		BLEU-3: 0.135355  BLEU-4: 0.084478
7)	Baseline+Attention +Bpe	BLEU-1: 0.201733  BLEU-2: 0.115805  BLEU-3: 0.102921  BLEU-4: 0.058725
8)	Baseline+Attention +GloVe+Bpe	BLEU-1: 0.228839  BLEU-2: 0.128127  BLEU-3: 0.113910  BLEU-4: 0.068561

BLEU-  
1: `corpus_bleu(actual,predicted,weights(1,0,0,0,0))`

This evaluation metric uses a weight of 1 for unigram and the rest of bigram, trigram and higher n-grams will be given a weight of 0

BLEU-  
2: `corpus_bleu(actual,predicted,weights(0.5,0.5,0,0,0))`

This evaluation metric uses a weight of 0.5 for unigram and bigram, the rest trigram and higher n-grams will be given a weight of 0

BLEU-  
3: `corpus_bleu(actual,predicted,weights(0.3,0.3,0.3,0,0))`

This evaluation metric uses a weight of 0.3 for unigram, bigram, trigram and higher n-grams will be given a weight of 0

BLEU-  
4: `corpus_bleu(actual,predicted,weights(0.25,0.25,0.25,0.25))`

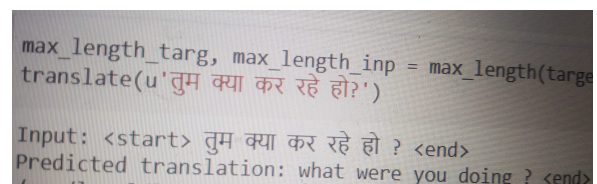
This evaluation metric uses a weight of 0.25 for unigram and bigram, trigram and 4-grams

All of our improvements outperformed the baseline model. Anyhow, a significant improvement can be observed when baseline is implemented with Attention and GloVe .

Implementation of GloVe along with the baseline model showed highest improvement in the performance without merging any other improvements

Using attention did improve the BLEU Score as it helps in alignment of the words in output language. But usage of GloVe, we take the advantage of having a pre-trained word embedding as it helps us to reduce the number of steps to converge to the output. BPE is expected to improve the BLEU score the best. But in our code, we initially encode on the input sentences using bpe model but we are using Keras word embedding. We think, this is causing the reduction in performance.

Here is a sample output



```

max_length_targ, max_length_inp = max_length(target)
translate(u'तुम क्या कर रहे हो?')

Input: <start> तुम क्या कर रहे हो ? <end>
Predicted translation: what were you doing ? <end>

```

## Conclusion:

In this project we have tried to implement the state-of-the-art sub word embedding technique and check how better it would perform compared to the traditional techniques used in neural machine translation. We think we weren't able to achieve the maximum performance with Byte-Pair Encoding because of two reasons.

First one being our dataset, which is small and there is high probability that most of the vocabulary in the test might have already been seen while training. Now this BPE is supposed to drastically improve the performance when test set contains many unknown words. Ans Second reason is, in our code we are splitting the words to sub words using `encode` in BPE but are using `kearas` to convert it to id and vector. Instead, we should have used `enocde_id` of BPE which could have improved the performance better than GloVe.

Best Performance of the model with GloVe shows us how important word Embeddings are and how much the pre-trained word embeddings are helpful.