SUDEEP CHOUDHARY (CS1727), M.Tech (Computer Science)

Unsupervised Machine Translation For Indian Languages Using Monolingual Corpora

Master's Thesis

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Technology (Computer Science)

submitted to

Indian Statistical Institute

Supervisor

Dr. Utpal Garain

Indian Statistical Institute

Kolkata, June 2019

Abstract

Machine translation has traditionally relied on parallel data but the amount of parallel data available for Indian languages is very less . The parallel data for Hindi-Marathi translation is around 50000 sentences which is very less in terms of data set required for supervised machine translation. But the good news is that monolingual data is very easy to find for this low-resource Indian languages .The aim of this project is to investigate whether it is possible to learn without the help of any parallel data . To serve the purpose we have implemented a model that takes sentences from two different monolingual corpora of different languages and maps them into the same latent space. We can encode sentences into the same latent space and can translate into any of the required languages . In this way, the model effectively learns to translate (encode/decode) without any form of supervision .The model only relies on monolingual corpora of two different languages and in our case it is Hindi and Marathi .The BLUE scores achieved by the model for Hindi to Marathi is 18.40 and Marathi to Hindi is 22.84 on the FIRE data set without using a single parallel sentence at training time.

Acknowledgements

Firstly, I would like to convey my deepest gratitude to my guide Prof. Utpal Garain, CVPR Unit, ISI Kolkata. I thank him for giving me ideas, criticizing and appreciating my efforts when necessary. I would also like to thank my senior and colleague Arjun Das, JRF, Calcutta University for bearing with me and helping me by giving important suggestions throughout the year. I would also like to thank ISI Kolkata, for providing me the opportunity to be acquainted with so many great minds, my teachers and my friends.I thank all of them for making me a part of the family. I thank my parents, whose constant support made me the person I am now.

Certification

This is to certify that this thesis titled **Unsupervised Machine Translation for Indian Languages Using Monolingual Corpora** submitted by **Sudeep Choudhary (CS1727)** , embodies the work done under my supervision.

Prof. Utpal Garain
CVPR Unit,
ISI Kolkata

Contents

List of Figures

1 Introduction

Because of natural ambiguity and diversity of human language their does not exist a single translation which can be considered as best translation . Because of this ambiguity language translation is still considered as one of the most difficult jobs in the field of artificial intelligence .
In the early 1950's machine translation was mostly rule based and relied completely on bilingual dictionaries. But with the emergence of neural networks , machine translation reached its new height.
Different approaches of Machine Translation:

- **Rule-Based Machine Translation (R.B.M.T) -** The most classical machine translation approach is rule-based machine translation. It is based on linguistic information about source and target languages retrieved from dictionaries and grammars covering the main semantic, morphological, and syntactic regularities of each language respectively.

- **Statistical Machine Translation (S.M.T)-** It uses statistical analysis and predictive algorithms to define rules that are best suited for target sentence translation. In case of S.M.T , a document is translated according to the probability distribution $p(t|s)$ where s is the source sentence and t is the translation of s.

- **Neural Machine Translation (N.M.T) -** It is an end to end system in which we use neural networks to learn the model for machine translation. Unlike the rule based models , N.M.T does not require huge amount of domain knowledge for translation.

1.1 Motivation for Unsupervised Neural Machine Translation

Supervised neural machine translation gives state-of-the-art results for most cases but it requires huge amount of parallel data .The amount of parallel data available for Indian languages such as Hindi-Marathi ,Hindi-Bengali,etc is very less and their is no significance on applying supervised neural machine translation on this small data set. But we can easily acquire monolingual corpus for this regional languages. But this monolingual corpus can be used for machine translation only if we can have a model which does not require any parallel data for training i.e. unsupervised machine translation .The model implemented does not require any form of supervision and can train only with the help of monolingual corpus .

1.2 Outline

The rest of this report is organized as follows.

- In **chapter 2**, we have discussed preliminaries such as Sequence-to-Sequence N.M.T , Metrics For Evaluating Machine Translation , Fast-Text , etc .
- In **chapter 3**, we have discussed some other attempts of unsupervised or semi-supervised translation system.
- In **chapter 4**, we have explained the unsupervised neural machine translation system .
- **Chapter 5** gives an overview of the dataset used both for training and testing .
- In **chapter 6**, we have briefly explained the experimental setup and the corresponding results achieved .
- The **last chapter** concludes the report and gives an overview regarding the future direction of the problem .

.

2 Preliminaries

2.1 Neural Machine Translation

Neural Machine Translation (N.M.T) is an end to end system in which we use neural networks to learn the model for machine translation. Unlike the rule based models , N.M.T does not require huge amount of domain knowledge for translation .

- ***Sequence-to-Sequence N.M.T***
  It is an encoder-decoder model .The problem with multilayer perceptron neural network is that it can't handle sentences of variable length. To avoid this problem seq-to-seq model is introduced.
  - *Encoder :* Encoder encodes sentences of variable length into a fixed size vector which is also known as context vector.Encoder consists of Recurrent Neural Network (R.N.N) or one of its variants and the reason of using R.N.N is that it can handle input sequences of varying length . Moreover , it also captures the sequence information of a sentence.
  - *Decoder :* The input for the decoder is the context vector generated by the encoder and output is the target language.The decoder also mainly consists of R.N.N or one of its variants such as Long Short Term Memory (L.S.T.M).

  The source sentence can be represented as s=(s1,s2,....sn) and the target sentence can be represented as t=(t1,t2,.....tn) .The probability of target sentence given the source sentence can be represented as $P(t|s)$. N.M.T models to maximize the conditional probability of generating the target sentence given the source sentence $P(t|s)$. Let C be the fixed length context vector generated by the encoder. Since decoder generates each

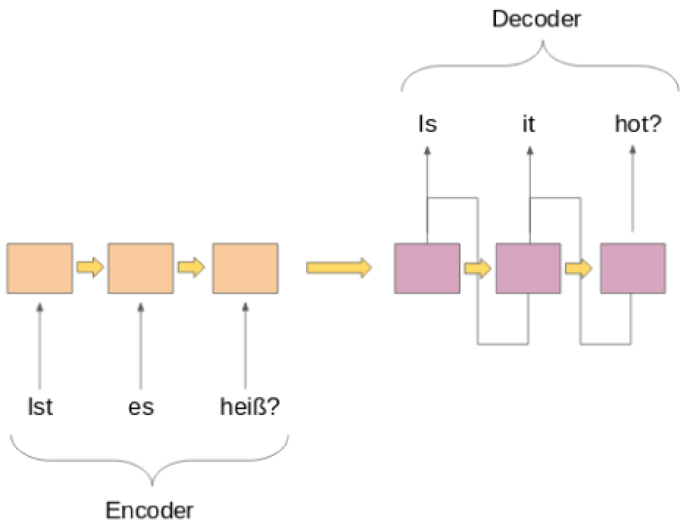


Figure 2.1: Representing a basic architecture of seq-to-seq N.M.T in which the source language is German and the target language is English. (source : analyticsvidhya.com)

word at each time step , therefore the conditional probability can be written as :

$$P(t|s) = \prod_{i=1}^{n} p(t_i|t < i, C) \tag{2.1}$$

taking log on both sides :

$$logP(t|s) = \sum_{i=1}^{n} logp(t_i|t < i, C) \tag{2.2}$$

The objective function of the N.M.T can be written as :

$$J_t = \sum_{t,s \in D} -logp(t|s) \tag{2.3}$$

- ***Sequence-to-Sequence N.M.T based on Attention Mechanism***
  The main problem of seq-to-seq N.M.T is long sentences and fixed length context vector . Since the dimension of the context vector is

same for both short and long sentences , it cannot capture entire information for long sentences . To avoid this attention mechanism is introduced which is currently the state-of-the-art on some benchmark problems of machine translation . Attention mechanism is based on the fact that when human translate a long sentence or a paragraph then first it read the entire sentence / paragraph and then it translate each word at a time by focusing on a particular part of the sentence/paragraph . Google Neural Machine Translation is also based on attention mechanism.

- Global Attention : Attends to all source words i.e. focusing on all source words while generating the translation of each word at a time.
- Local Attention : Attends to a specific window i.e. focusing on specific words while generating the translation each word at a time.

2.2 Metrics For Evaluating Machine Translation

At the end of every machine translation, the most important part will be to judge that which translation is the best. Human language is naturally ambiguous and diverse and their can be different translation for the same sentence each translation is correct . So, the metrics for evaluating machine translation is still an open problem .

Evaluation metrics should consider two points while evaluating a translator that the output generated is both syntactically and semantically correct and have the same meaning as of the reference sentence . The most common and significant metrics for evaluating machine translation are mentioned below.

- **Precision and Recall of words**
  Let us try to explain both precision and recall with the help of an example .
  Reference sentence : Ram goes to school daily.
  Output sentence : Everyday Ram go to school .

Precision = number of matched words / output-length
Precision = 3 / 5 = 60 percent
Matched words are Ram , to and school .

Recall = number of matched words / reference-length
Precision = 3 / 5 = 60 percent

Reference-length and output-length is same in this case which is equal to 5 in this case .

- **Word Error Rate**

  - matched : number of words matched in the output sentence.
  - deleted : number of words deleted in the output sentence.
  - inserted : number of words inserted in the output sentence.
  - substituted : number of words substituted in the output sentence.

  Word Error Rate = (deleted + inserted + substituted) / reference-length
- **Bilingual Evaluation Understudy Score (B.L.E.U)**
  B.L.E.U is considered to be the most significant and dominant metric in terms of machine translation. The B.L.E.U score was proposed by Kishore Papineni, et al. in their 2002 paper "B.L.E.U: a Method for Automatic Evaluation of Machine Translation".
  B.L.E.U score ranges from 0 to 1 where 1 being the best score that is both the reference and output sentence is a complete match. Sometimes it is multiplied by 100 , in that case it will range from 0 to 100, where 100 being the best .
  key points of B.L.E.U score are mentioned below :

  - It looks for n-gram matches between the reference and the output sentence .
  - It computes precision for n-grams of size 1 to 4 .
  - It adds penalty for short translations which is also known as brevity penalty .
  - It is generally computed over each and every singe sentence of the corpus .

$$B.L.E.U = min(1, output\ length/reference\ length)\ (\prod_{i=1}^{4} precision_i)^1/4 \tag{2.4}$$

Reference sentence : Ram goes to school daily.
Output sentence : Everyday Ram goes to school .

Let us try to calculate the B.L.E.U score of the above example .

precision (1-gram)	4/5
precision (2-gram)	3/4
precision (3-gram)	2/3
precision (4-gram)	1/4
B.L.E.U score	.67

But one of the major drawbacks of B.L.E.U score is that it does not consider synonyms i.e. it will not consider the words having same meaning at the time precision calculation .Moreover , it should also give partial credit to stem words.

- **METEOR**
  It handles some of the drawbacks of B.L.E.U score as it gives partial credit to the words having same meaning . Moreover, it also consider stem words and give some credit for matching stems. It consider both precision and recall.

2.3 FastText

When we deal with any N.L.P task , the most important thing is to convert word into vector (word-embedding) . Word2Vec is commonly used to generate word embedding but one of the problem of Word2Vec is that it can't handle rare words. FastText developed by Facebook can handle this problem. It breaks word into several n-gram(sub-words) . For example , apple can be broken as app , ppl and ple in case of 3-gram .The word embedding generated will be weighted sum of all these n-grams .Rare words can now be properly represented since it is highly likely that some of their n-grams also appears in other words.

3 Related Works

One of the main drawbacks of deep learning methods is that it requires huge amount of data and in case of supervised neural machine translation huge amount of parallel data is required . To avoid this there have been several attempts to use monolingual corpus which is easily available even for low-resource languages .

(Sennrich, Haddow, and Birch, 2016) proposed a method in which a dummy model is initially trained on the available parallel data and then that model is used to produce translations of a large monolingual corpus and the produced translation of the monolingual data is merged in the target side. Finally , this merged data is used to train the original translation system. But in this case we are still dependent on the parallel data for the initial development of auxiliary model.

(Gülçehre et al., 2015) proposed a method in which the model is trained with the parallel data and then the decoder is augmented with a language model by attaching the monolingual corpus on the target side . Initially , the neural machine translator is trained on parallel corpus and the language model is trained on monolingual corpus separately . But , this model also requires some amount of parallel data .

(Pourdamghani and Knight, 2017) proposed a method in which it converts the problem of machine translation into a cipher problem . This model can be considered as a zero-parallel-resource machine translation but it only works for short sentences and also requires closely related languages.

4 Unsupervised Neural Machine Translation

4.1 Method Overview

Unsupervised Neural Machine Translation proposed by (Lample, Conneau, Denoyer, et al., 2018) is a zero-parallel-resource machine translation . It only assumes that there exists a monolingual corpus for each language .

The key idea of this method is based on three principles :

- Model has to reconstruct a sentence given a noisy version of it (Vincent et al., 2008) which is also known as denoising auto-encoders . Let x be a sentence of source and C(x) be a noisy version of it. C(x) is the input for encoder and it generates Z_{src}. Z_{src} is feed into decoder and it reconstructs x as $\hat{x}$.
- Model is trained to reconstruct any source sentence given a noisy version of it in the target domain (Sennrich, Haddow, and Birch, 2016). Let x is the source sentence and y is the noisy translation generated by the model M in its previous iteration t , y = M^t(x) . C(y) is the noisy

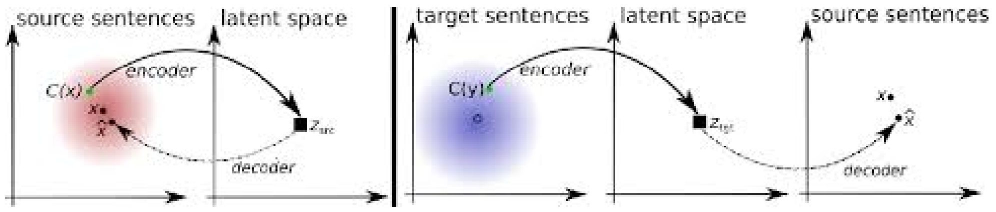


Figure 4.1: Left : Auto Encoder Right : Translation . source : (Lample, Conneau, Denoyer, et al., 2018)

version of y . C(y) is the input for encoder and it generates Z_{tgt}. Z_{tgt} is fed into decoder and it produces a noisy translation $\hat{x}$.

- To build a common latent space between the source and the target language . Both the languages are forced to have same distribution using adversarial training (Ganin et al., 2016) . Encoder is trained to fool the discriminator and the discriminator is trained to identify sentence representation correctly .

To keep the model fully unsupervised , FastText (Bojanowski et al., 2017) is used to initialize the model . The input for FastText is concatenated monolingual corpus of Hindi and Marathi. Since , in our case the source and the target language is similar and have lot of overlapping words , FastText can be used to generate word by word translation of sentences .

4.2 Architecture of the Translation System

The translation system consists of an encoder and a decoder . Encoder encodes sentences to a latent space and from that latent space decoder decodes sentences into source or target language .
W_s be the set of all words in the source domain and W_t be the set of all all words in the Target domain. Z_s is the set of embedding generated by FastText (Bojanowski et al., 2017) in the source domain and Z_t is the set of embedding generated by FastText in the target domain . Then ,

$$Z_s = \left(z_1^s, z_2^s, \ldots, z_{|W_s|}^s\right)$$
$$Z_t = \left(z_1^t, z_2^t, \ldots, z_{|W_t|}^t\right)$$

Let the input sentence x consists of m words x=$(x_1, x_2, \ldots, x_m)$ in a particular language l $\in$ (l_1, l_2) . x is the input of encoder and it generates a sequence of hidden states z= $(z_1, z_2, \ldots, z_m)$ corresponding to the word embeddings . Therefore , the input for encoder is x (word embedding of a sentence) and language l . Encoder can be denoted as e(x,l) and Θ_{enc} are the parameters of encoder shared between the source and the target language . Sequence-to-sequence model with attention (Bahdanau et al., 2015) is used . Encoder

is a bi-directional L.S.T.M where the input is a sequence of words and the output is a sequence of hidden states .
The input for decoder is the sequence of hidden states z= $(z_1, z_2, \ldots, z_m)$ and language l. Decoder can be denoted as d(z ,l) and it generates a sequence of words y= $(y_1, y_2, \ldots, y_k)$ in the source or the target domain . Θ_{dec} are the parameters of decoder shared between the source and the target language. Decoder is a L.S.T.M and at each time , the decoder takes as input the previous hidden state, the current word and a context vector given by a weighted sum over the encoder states and generate $y_i^{'}$s .

4.3 DeNoising Auto-Encoders

If a sequence-to-sequence model with attention mechanism is used to auto-encode sentence , then it won't make sense ,since it will only learn to copy data word by word. It won't learn any structure of the data . To avoid this, strategy of Denoising Auto-encoders (DAE) (Vincent et al., 2008), is used . A stochastic noise model denoted as C , which operates on sentence is used to generate a noisy version of it .
Noise Model : A noise is induced into the sentence by two means .

- Each word is deleted with probability p$_{wd}$. p$_{wd}$ is a tuning parameter and in our case , p$_{wd}$ = 0.3 .
- We slightly shuffle the input sentence . To serve the purpose , a permutation function σ is applied to the input sentence such that $\forall i \in i, n$ where n is the length of input sentence , $|\sigma(i) - i| <= k$ i.e. each word can atmost be K distance away from its original position. K is again the tuning parameter and in our case , k = 3 .

Let us try to define the process of DeNoising Auto-Encoder :

- Let x be a sentence which belongs to either language i.e. source or target , x ∈ l.
- A stochastic noise model C operates on sentence x and it produces C(x) .
- The input for encoder is C(x) and it can be represented as e(C(x) ,l) .
- Decoder reconstructs x as $\hat{x}$ and it is represented as d(e(C(x) ,l) ,l) .

Therefore , the objective function can be defined as follows :

$$L_{auto}(\Theta_{enc}, \Theta_{dec}, Z, l) = E_{x \sim D_l, \hat{x} \sim d(e(C(x),l),l)}[\Delta(x, \hat{x})] \tag{4.1}$$

In this equation, Δ is a measure of discrepancy between the two sequences, the sum of token-level cross-entropy losses in our case .

4.4 Cross Domain Training

The ultimate goal is to translate sentence from the source/target domain to target/source domain . Let us define the process of Cross Domain Training :

- Let x $\in$ D_{l_1} . By applying the current translation model M , it generates y = M(x) , y $\in$ D_{l_2} .
- C(y) is a randomly sampled noisy version of y. Now , the encoder will look like e(c(M(x)), l_2) .
- Decoder translates from the given input and it reconstructs x as $\hat{x}$. It can be represented as d(e(c(M(x)), l_2), l_1) .

The objective is thus to learn the encoder and the decoder such that they can reconstruct x from C(y).The cross domain loss can be written as :

$$L_{cross}(\Theta_{enc}, \Theta_{dec}, Z, l_1, l_2) = E_{x \sim D_{l_1}, \hat{x} \sim d(e(c(M(x)),l_2),l_1)}[\Delta(x, \hat{x})]$$

(4.2)

Again Δ is a measure of discrepancy between the two sequences, the sum of token-level cross-entropy losses in our case .

4.5 Adversarial Training

The decoder of a neural machine translation works well only if the output features of encoder is in the same space regardless of the actual language of the input sentence . To serve the purpose , we train a neural network, which we will refer to as the discriminator, to classify between the encoding of source sentences and the encoding of target sentences (Ganin et al., 2016). The discriminator is designed to classify the languages correctly and the encoder tries to fool the discriminator such that it can't classify correctly . The input for discriminator is output of encoder , which is a sequence of m hidden states (vector) z=$(z_1, z_2, \ldots, z_m)$, with $z_i \in R^n$. Discriminator predicts with probability $p_D(l|z_1, z_2, \ldots, z_m) \propto \prod_{j=1}^{m} p_D(l|z_j)$, where 0 corresponds to the source domain and 1 corresponds to the target domain. The discriminator is trained to predict the language by minimizing the following cross-entropy loss:

$$L_D(\Theta_D|\Theta, Z) = -E_{(x_i,l_i)}[log p_D(l_i|e(x_i, l_i)] \tag{4.3}$$

where Z are the encoder word embedding , Θ_D are the parameters of Discriminator and (x$_i$, l$_i$) corresponds to sentence and language id pairs .

The encoder is trained to fool the discriminator i.e. the discriminator should predict incorrect classification .

$$L_{adv}(\Theta_{enc}, Z|\Theta_D) = -E_{(x_i,l_i)}[log p_D(l_j|e(x_i, l_i)] \tag{4.4}$$

with $l_j = l_1$ if $l_i = l_2$, and vice versa and Θ_{enc} are the parameters of encoder.

4.6 Final Objective Function

The final objective function will be the sum of equation(2.1) , equation(2.2) and equation(2.4) . The discriminator will be trained in parallel with the loss function of equation(2.3) keeping the parameters of encoder fixed . The final objective function is :

$$\begin{aligned} L(\Theta_{enc}, \Theta_{dec}, Z) =& \lambda_{auto}|L_{auto}(\Theta_{enc}, \Theta_{dec}, Z, src)| + \lambda_{auto}|L_{auto}(\Theta_{enc}, \Theta_{dec}, Z, tgt)| \\ &+ \lambda_{cross}|L_{cross}(\Theta_{enc}, \Theta_{dec}, Z, src, tgt)| + \lambda_{auto}|L_{auto}(\Theta_{enc}, \Theta_{dec}, Z, tgt, src)| \\ &+ \lambda_{adv}|L_{adv}(\Theta_{enc}, Z|\Theta_D)| \end{aligned}$$

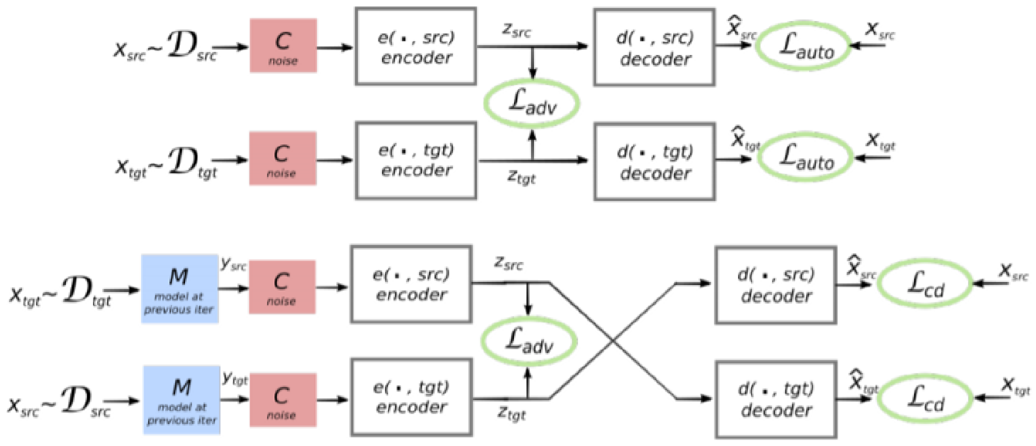


Figure 4.2: Training Objectives Top : Auto Encoder , Bottom : Translation. source : (Lample, Conneau, Denoyer, et al., 2018)

where λ_{auto} , λ_{cross} and λ_{adv} are hyper-parameters determining the significance of auto-encoding , cross domain and adversarial training .Auto-encoder and cross domain training is done on both sides i.e source and target domain .

5 Dataset

In this experiment , Hindi-Marathi language pair of F.I.R.E dataset is considered where Hindi is the source language and Marathi is the target language. For the training purpose , monolingual corpus is used and for validation and testing purpose parallel corpus is used.
Since there does not exist any kind of correspondence between Hindi and the Marathi monolingual dataset , so we can claim that the process does not depend on any parallel data .We lower-case the entire data set and removed sentences having more than 50 words .
Table 5.1 gives an overview of the training data . Hindi monolingual corpus consists of 60609002 words and 3548000 sentences. Marathi monolingual corpus consists of 27763210 words and 2798161 sentences .
Table 5.2 gives an overview of the validation data. Hindi parallel corpus consists of 17535 words and 1000 sentences. Marathi parallel corpus consists of 13178 words and 1000 sentences .
Table 5.3 gives an overview of the test data. Hindi parallel corpus consists of 24115 words and 1000 sentences. Marathi parallel corpus consists of 23873 words and 1000 sentences .

	word count	sentence count
Hindi Monolingual Corpus	60609002	3548000
Marathi Monolingual Corpus	27763210	2798161

Table 5.1: Details of Training data

	word count	sentence count
Hindi Parallel Corpus	17535	1000
Marathi Parallel Corpus	13178	1000

Table 5.2: Details of Validation Data

	word count	sentence count
Hindi Parallel Corpus	24115	1000
Marathi Parallel Corpus	23873	1000

Table 5.3: Details of Test Data

6 Results and Experiments

6.1 Bilingual Dictionary

To keep the entire process unsupervised, we need to generate a bilingual dictionary to initialize the translation model. Since both the languages Hindi and Marathi have lot of word overlapping , both Hindi and Marathi monolingual corpus is randomly shuffled and then concatenated to produce a combined corpus. Then this concatenated corpus is fed into fastText (Bojanowski et al., 2017) to generate word embedding (word vector) . This bilingual dictionary generated is used for initial word-by-word by translation.

6.2 Experimental Details

Discriminator

- It is a multi layer perceptron with three hidden layers each of size 1024.
- Leaky-ReLU is used as an activation function.
- Smoothing coefficient of the discriminator is kept as , s=0.1 .
- It is trained using RMSProp (Salakhutdinov and Hinton, 2012) with a learning rate of 0.0005 .

Training Parameters

- The encoder and the decoder are trained using Adam (Kingma and Ba, 2014).

- The learning rate is 0.0003 and mini batch-size = 16 .
- The process of training alternates between encoder-decoder and discriminator .

6.3 Results

Table 6.1 shows B.L.E.U score of validation data after every epoch. For (Hindi-Marathi) , 18.41 is the best B.L.E.U score and for (Marathi-Hindi) , 22.74 is the best B.L.E.U score achieved after 24 epochs.

Validation Data	(Hindi- Marathi)	(Marathi- Hindi)
Epoch 0	4.64	6.00
Epoch 1	7.74	12.29
Epoch 2	12.13	16.37
Epoch 3	13.19	18.27
Epoch 4	14.73	18.60
Epoch 5	15.15	18.67
Epoch 7	15.30	19.85
Epoch 9	15.84	20.14
Epoch 10	16.02	20.90
Epoch 16	17.36	23.08
Epoch 24	18.41	22.74

Table 6.1: B.L.E.U score of validation data on FIRE data set

Table 6.2 shows B.L.E.U score of both (Hindi-Marathi) and (Marathi-Hindi) of test data after every epoch. For (Hindi-Marathi) , 18.40 and for (Marathi-Hindi) , 22.84 is the best B.L.E.U score achieved after 24 epochs.
Figure 6.1 shows the graphical representation of B.L.E.U score with respect to number of iterations(epoch) for validation data .
Figure 6.2 shows the graphical representation of B.L.E.U score with respect to number of iterations(epoch) for test data .
In figure 6.3 , iteration 0 corresponds to word-by-word translation produced by FastText . After 24 iterations, model generates very good translations , very close to reference text .

Test Data	(Hindi- Marathi)	(Marathi- Hindi)
Epoch 0	4.36	5.93
Epoch 1	7.91	12.34
Epoch 2	12.32	16.01
Epoch 3	13.42	17.80
Epoch 4	14.93	18.89
Epoch 5	15.50	18.67
Epoch 7	16.27	19.75
Epoch 9	15.97	20.95
Epoch 10	17.27	20.75
Epoch 16	18.12	22.63
Epoch 24	18.40	22.84

Table 6.2: B.L.E.U score of test data on FIRE data set

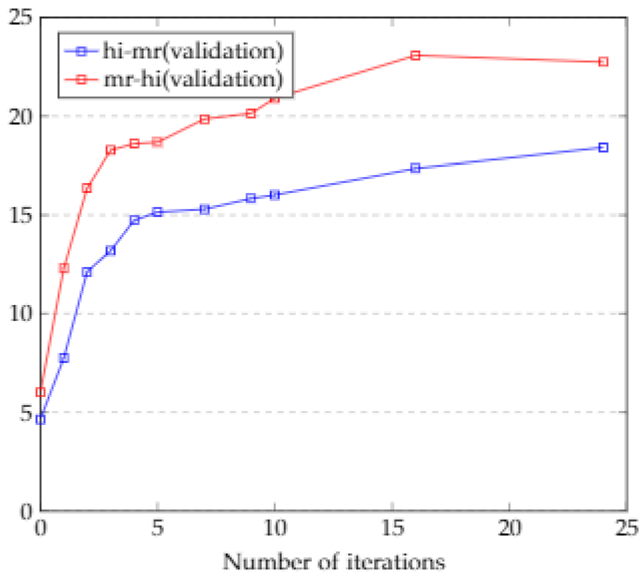


Figure 6.1: B.L.E.U score of validation data with respect to number of iterations

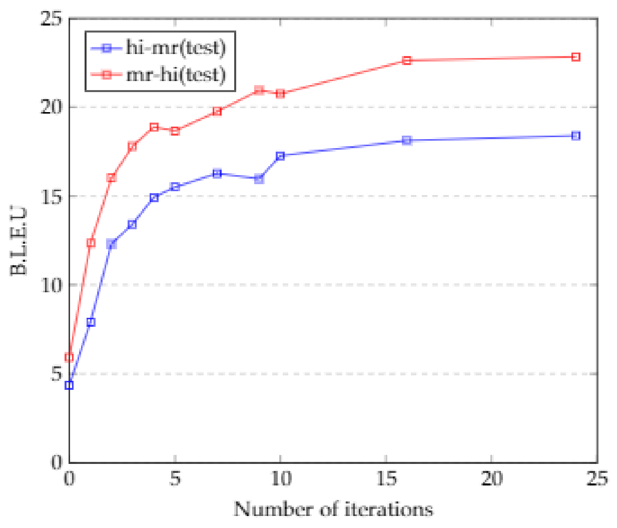
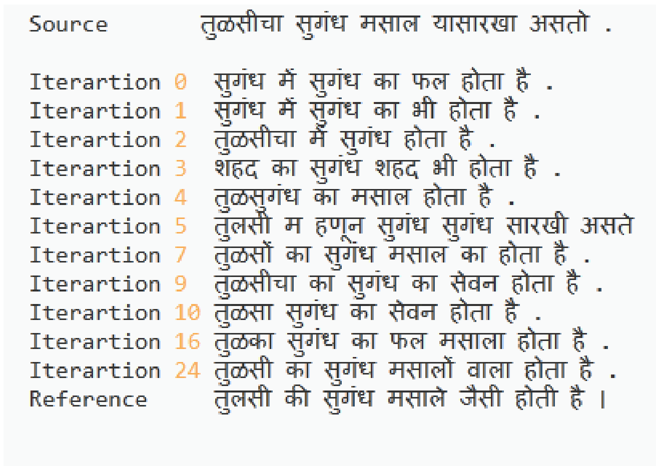


Figure 6.2: B.L.E.U score of test data with respect to number of iterations



```
Source         तुळसीचा सुगंध मसाल यासारखा असतो .

Iterartion 0  सुगंध में सुगंध का फल होता है .
Iterartion 1  सुगंध में सुगंध का भी होता है .
Iterartion 2  तुळसीचा में सुगंध होता है .
Iterartion 3  शहद का सुगंध शहद भी होता है .
Iterartion 4  तुळसुगंध का मसाल होता है .
Iterartion 5  तुलसी म हणून सुगंध सुगंध सारखी असते
Iterartion 7  तुळसों का सुगंध मसाल का होता है .
Iterartion 9  तुळसीचा का सुगंध का सेवन होता है .
Iterartion 10 तुळसा सुगंध का सेवन होता है .
Iterartion 16 तुळका सुगंध का फल मसाला होता है .
Iterartion 24 तुळसी का सुगंध मसालों वाला होता है .
Reference     तुलसी की सुगंध मसाले जैसी होती है ।
```

Figure 6.3: Example of translations on the Marathi-Hindi pair of the F.I.R.E dataset after every iteration .

6.4 Discussion

Moses is a well known statistical machine translation system . The B.L.E.U score generated by Moses for (Marathi-Hindi) translation on the test data is

32.40 and in our case it is 22.84 . It is bit higher than the result generated by our unsupervised machine translation system because of the following reasons :

- The test dataset mostly belongs to tourism or health domain . Moses was also trained on a dataset which belongs to tourism and health domain .Our training dataset belongs to general domain and this can be a reason for that difference in B.L.E.U score .
- Lack of availability of parallel data in general domain for (Marathi-Hindi) restricts us to do a fair comparison between the unsupervised N.M.T and Moses .
- The model implemented is an unsupervised model and it can be considered as a good starting point to explore the scope of unsupervised model for Indian languages .

With the availability of test data in general domain in near future will allow us to get a better idea that how this system works for Indian languages. Since it is an unsupervised model , so we can try it for many other Indian languages such as (Hindi-Bengali), (Hindi-Urdu), (Hindi-Tamil), etc.

7 Conclusion and Future Work

We implemented a translation model for Hindi-Marathi language pair which is learned using monolingual datasets only, without any alignment between sentences or documents . But for the initialisation of our model, we need to generate a bilingual dictionary .For Hindi-Marathi language pair , FastText (Bojanowski et al., 2017) is used to generate bilingual dictionary but this works only if the language pair had lot of overlapping words.Fortunately , Hindi-Marathi language pair has lots of common words but language pair such as English-Hindi , Hindi-Bengali ,etc does not have overlapping words. So in this case FastText (Bojanowski et al., 2017) can't be used to generate bilingual dictionary. So, this model fails in case of language pairs that does not have overlapping words.
We need to generate a bilingual dictionary in an unsupervised way to initialise the translation model. Word translation without parallel data (Lample, Conneau, Ranzato, et al., 2018) can be used to generate a simple unsupervised word-by-word translation model, and to improve the model using DeNoising Auto Encoder (Vincent et al., 2008), Cross domain trainig and a discriminator to align latent space distributions (Lample, Conneau, Denoyer, et al., 2018). This can be a solution for language pairs which does not have overlapping words .

Bibliography

Bojanowski, Piotr et al. (2017). "Enriching Word Vectors with Subword Information." In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146 (cit. on pp. 12, 19, 25).

Ganin, Yaroslav et al. (2016). "Domain-adversarial Training of Neural Networks." In: *J. Mach. Learn. Res.* 17.1, pp. 2096–2030. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=2946645.2946704 (cit. on pp. 12, 15).

Gülçehre, Çaglar et al. (2015). "On Using Monolingual Corpora in Neural Machine Translation." In: *CoRR* abs/1503.03535. arXiv: 1503.03535. URL: http://arxiv.org/abs/1503.03535 (cit. on p. 9).

Kingma, Diederik P. and Jimmy Ba (2014). *Adam: A Method for Stochastic Optimization*. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015. URL: http://arxiv.org/abs/1412.6980 (cit. on p. 19).

Lample, Guillaume, Alexis Conneau, Ludovic Denoyer, et al. (2018). "Unsupervised machine translation using monolingual corpora only." In: *International Conference on Learning Representations (ICLR)* (cit. on pp. 11, 16, 25).

Lample, Guillaume, Alexis Conneau, Marc'Aurelio Ranzato, et al. (2018). "Word translation without parallel data." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=H196sainb (cit. on p. 25).

Pourdamghani, Nima and Kevin Knight (2017). "Deciphering Related Languages." In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. Copenhagen, Denmark: Association for Computational Linguistics, 2513–2518. URL: https://www.aclweb.org/anthology/D17-1266 (cit. on p. 9).

Salakhutdinov, Ruslan and Geoffrey Hinton (2012). "An Efficient Learning Procedure for Deep Boltzmann Machines." In: *Neural Comput.* 24.8, pp. 1967–2006. ISSN: 0899-7667. DOI: 10.1162/NECO_a_00311. URL: http://dx.doi.org/10.1162/NECO_a_00311 (cit. on p. 19).

Sennrich, Rico, Barry Haddow, and Alexandra Birch (2016). "Improving Neural Machine Translation Models with Monolingual Data." In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 86–96. DOI: 10.18653/v1/P16-1009. URL: https://www.aclweb.org/anthology/P16-1009 (cit. on pp. 9, 11).

Vincent, Pascal et al. (2008). "Extracting and composing robust features with denoising autoencoders." In: *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, pp. 1096–1103. DOI: 10.1145/1390156.1390294. URL: https://doi.org/10.1145/1390156.1390294 (cit. on pp. 11, 13, 25).