Automatic Text Translation of Multilingual Sentences using Transformer

by

EDARA VEERA VENKATA HARI CHARAN
202011056

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

MASTER OF TECHNOLOGY
in
INFORMATION AND COMMUNICATION TECHNOLOGY
to

DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

May, 2022

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Master of Technology in Information and Communication Technology at Dhirubhai Ambani Institute of Information and Communication Technology and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

[illegible]

Edara Veera Venkata Hari Charan

Certificate

This is to certify that the thesis work entitled AUTOMATIC TEXT TRANSLATION OF MULTILINGUAL SENTENCES USING TRANSFORMER has been carried out by EDARA VEERA VENKATA HARI CHARAN for the degree of Master of Technology in Information and Communication Technology at *Dhirubhai Ambani Institute of Information and Communication Technology* under our supervision.

[illegible] 29/07/22

Prof.Manjunath V. Joshi
Thesis Supervisor

[illegible]

Avik Hati
Thesis Co-Supervisor

Acknowledgments

I would like to express my sincere gratitude to my research supervisor, Professor Manjunath V. Joshi, Dhirubhai Ambani Institute of Information and Communication Technology for giving me the opportunity to work in the area of Natural language processing and providing his invaluable guidance throughout the course of this work. He helped and motivated me to explore new ideas throughout the course of the research. I am also very grateful to my co-supervisor, Professor Avik Hati, Dhirubhai Ambani Institute of Information and Communication Technology for guiding me in the right direction and correcting me at every small-step throughout the journey. It was a privilege and honor to work and study under their guidance.

Special mention to my peer, Makhija Shradha, pursuing MTech at Dhirubhai Ambani Institute of Information and Communication Technology for being part of the discussions and the experimentation throughout the research work. I would also like to thank my institute for providing me with the necessary resources for the research.

Last but not the least, I would like to thank my family and friends for their help and support in these unprecedented times.

Contents

Abstract

Machine translation from one language to another is a complex problem in machine learning and one in which the machine still cannot achieve satisfactory results. The recent focus for solving this challenge has been on neural machine translation (NMT) techniques, by using architectures such as recurrent neural network (RNN) and long term short memory (LSTM). But the architecture of transformer is able to outperform these NMT techniques. The architecture of the transformer has been successfully utilized to build models that target a single language pair translation or translation among multiple languages. But it currently lacks research in the area of translation of multilingual sentences, where each sentence is in the form of a mixture of languages. In this work we will establish a model based on the transformer architecture that can translate multilingual sentences into a single language, with the help of a multilingual neural machine translation (MNMT) model and custom made datasets.

List of Acronyms

BLEU Bilingual Evaluation Understudy Score

GPU Graphics Processing Unit

HKD Hierarchical Knowledge Distillation

HMT Hybrid Machine Translation

LSTM Long Short-Term Memory

MNMT Multilingual Neural Machine Translation

NMT Neural Machine Translation

RBMT Rule Based Machine Translation

RELU Rectified Linear Unit

RNN Recurrent Neural Network

Seq2Seq Sequence to Sequence

SMT Statistical Machine Translation

List of Tables

List of Figures

CHAPTER 1

Introduction

The task of machine translation was originally performed using techniques that involved dictionary matching [23]. Then the approach was changed into a more rule oriented manner. In the past few decades most translations models were based upon the statistical machine translation approach [14]. In this type of model, the translation process depends on units such as phrases and sentences. Group of one or more words constitute a phrase. The goal of these models is to estimate the probability of translation for a pair of phrases. But the pairs consist of one phrase from one language and the other phrase from another language. In an SMT model the process of pairing the correct pair and predicting the correct pair is very hard as the probability of these phrases occurring in the training data is less. Increasing the size of the training data provided better results, but they were still not satisfactory. Hence this creates a requirement of alternatives for the purpose of machine translation.

Recently organizations like Google and Bing have shifted their focus in the research for translation towards NMT. The memory requirement of these NMT models is only a fraction of that of the more conventional SMT models. To maximize the translation performance every part of the NMT model is trained in a joint manner as compared to the more conventional translation systems [27]. Problems such as machine translation fall under the category of sequential modeling and transduction problems. Many solidly formed approaches using RNN [19], LSTM [10] and gated recurrent neural networks [5] already exist.

These models are recurrent in nature and generally consider the positions of symbols that exist both in the input sequences and the output sequences. Hidden states h_t are generated by taking the earlier hidden state h_{t-1} as well as the input for position t. These hidden states are generated by lining up the positions with respect to the steps in the process of computation. The sequential nature displayed in these problems makes it impossible for any parallelization within the training examples. As the length of the sentences increases, this issue becomes more critical in nature. There have been many efforts to deal with this issue using various techniques in recurrent language models and encoder-decoder architectures [11] [18].

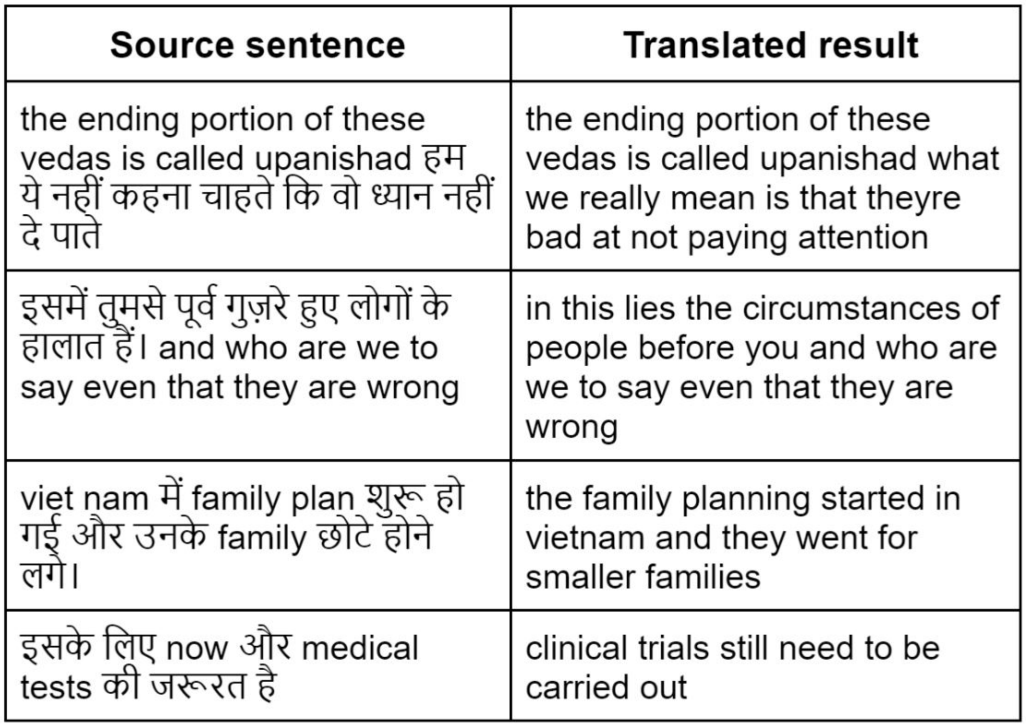
It can be said that mechanisms that make use of attention based approaches have become vital while dealing with sequence modeling and transduction models as they allow us to model the dependencies without focusing too much on how far apart are they positioned in the input and output sequences [2]. This requires a sequence to sequence architecture that can utilize the mechanism of self attention to assign varying significance to different parts of the input data, which brings the transformer into light.

Many NMT models generally target a single language pair translation, where the input is in the form of one language and the generated translated output is in the desired language. Whereas models like multilingual neural machine translation MNMT [24], utilize a single NMT model to enable translation among multiple languages instead of training a single model for every individual language pair. But little research has been done on the ability of NMT models to translate multilingual sentences, where the input sentence may consist of entire phrases or words in multiple languages, into the desired target language. This kind of translation can be useful in the translation of Lok sabha debate sessions that are recorded in two different languages. It can also be used to translate casual conversations often found in many messenger applications. This thesis tries to establish the ability of an MNMT model based on the architecture of a transformer to translate sentences consisting of multilingual phrases or words into a single language.

Problem statement - Given a dataset that has sentences composed of multiple languages, the aim is to translate these sentences into a single target language utilizing the transformer as shown in Fig. 1.1.

The major contributions of this work are as follows :

- A multilingual model is trained to translate sentences formed by the combination of two languages.
- Two custom datasets are created for the purpose of learning a transformer based multilingual model.



Source sentence	Translated result
the ending portion of these vedas is called upanishad हम ये नहीं कहना चाहते कि वो ध्यान नहीं दे पाते	the ending portion of these vedas is called upanishad what we really mean is that theyre bad at not paying attention
इसमें तुमसे पूर्व गुज़रे हुए लोगों के हालात हैं। and who are we to say even that they are wrong	in this lies the circumstances of people before you and who are we to say even that they are wrong
viet nam में family plan शुरू हो गई और उनके family छोटे होने लगे।	the family planning started in vietnam and they went for smaller families
इसके लिए now और medical tests की जरूरत है	clinical trials still need to be carried out

Figure 1.1: Illustrations of expected translations

CHAPTER 2
Literature Survey

We will now look into the types of machine learning techniques explored in this work. Firstly, we come across statistical machine translation (SMT) which was developed during the 1980s. The concept of SMT was designed while thinking of translation as a machine learning task. Provided with a large parallel corpus of already translated text, the job of the SMT is to take a series of symbols provided in the source language with its corresponding vocabulary and then convert them into a series of symbols in the target language formed with its corresponding vocabulary. The work in [16] explains the basic idea behind SMT and also sheds light on the challenges faced. It also shows a classification of the various methods in this area.

Then we look into rule-based machine translation (RBMT) which was first introduced in 1985. The design of RBMT is formulated on the definition of rules for syntax, and morphology of a language. In RBMT, a collection of rules and the vocabulary of both the source and target languages are required as the resources. The work in [25] explains the basic idea behind RBMT and shows a comparison between the statistical and rule-based approaches to machine translation with the help of a case study from the perspective of Indian languages.

The approaches mentioned so far have failed to attain an adequate level of accuracy individually. This led to the creation of hybrid machine translation (HMT). The chief principle behind HMT is that it combines and uses various machine translation methods in a single system. The most frequently used combination is that of SMT and RBMT. The work in [6] explains some of the popular hybridization methods and how they try and integrate the principal attributes of the various individual techniques while throwing light on the applications of these HMT methods.

If a machine translation system makes use of an artificial neural network, which leads to a better fluency of the machine translation and also improves its accuracy then it can be classified as a NMT model. These NMT models are generally established on the basis of an encoder and decoder structure and make use of RNN. The cyclic nature of RNN enables it to learn the repeated sequences much more efficiently as compared to other networks. In models such as ConvS2S [7], Extended Neural GPU [12] and ByteNet [13], conventional neural networks are used as the building blocks. The goal of all these models is to compute the hidden representation simultaneously for all input and output positions.

The problem with the above mentioned models is that the amount of computational resources consumed to establish a relationship between two arbitrary signals in the input or output positions increases proportionally to the separation between their positions. In the case of ConvS2S, it increases linearly and in the case of ByteNet, it increases logarithmically. This poses a challenge as learning the dependencies between positions [9] that are far apart becomes difficult. Learning these dependencies is a crucial aspect of any machine translations model.

By making use of the transformer, these issues are mitigated and the number of operations required to learn the dependencies between distant positions become constant and do not depend on the distance between two such positions. For the computation of the representation of a sequence, the concept of self-attention is utilized. It can also be called as intra-attention. It is used to relate the diverse positions of a single sequence. The concept of self-attention has been successfully utilized in performing various jobs, which include abstractive summarization [22], reading comprehension [4], learning task-independent sentence representations [15] and textual entailment [21]. Keeping this in mind we will now look at the architecture of the transformer in more detail in the next chapter.

Apart from the traditional NMT approaches a multilingual approach of the same translation method exists. It is multilingual neural machine translation (MNMT), in which the model is trained to facilitate translation between multiple languages instead of building different models for each individual language pair. This allows the model to train better on low-resource languages by using data from various other languages. The work in [24] shows how the model of an MNMT model is extremely dependent on the kind of languages utilized during training because of negative transfer which degrades the performance of the translation while transferring information from a set of various languages. It also proposes a technique for MNMT known as the hierarchical knowledge distillation (HKD) [24] to counter the effect of negative transfer.

CHAPTER 3

Transformer in NMT

The machine translation model follows an encoder-decoder structure as shown in Fig. 3.1. The job of the encoder is to map an input sequence $(a_1, ..., a_n)$ where a_i signifies a word embedding onto $c = (c_1, ..., c_n)$, where c_i is a representation of a word assigned to it by the encoder and c is a sequence of continuous representations. It is used by the decoder to generate a sequence $(b_1, ..., b_n)$ as output of words one word at a time. The nature of the model is auto regressive. The symbols that are generated previously are taken as input for generating the next symbol.

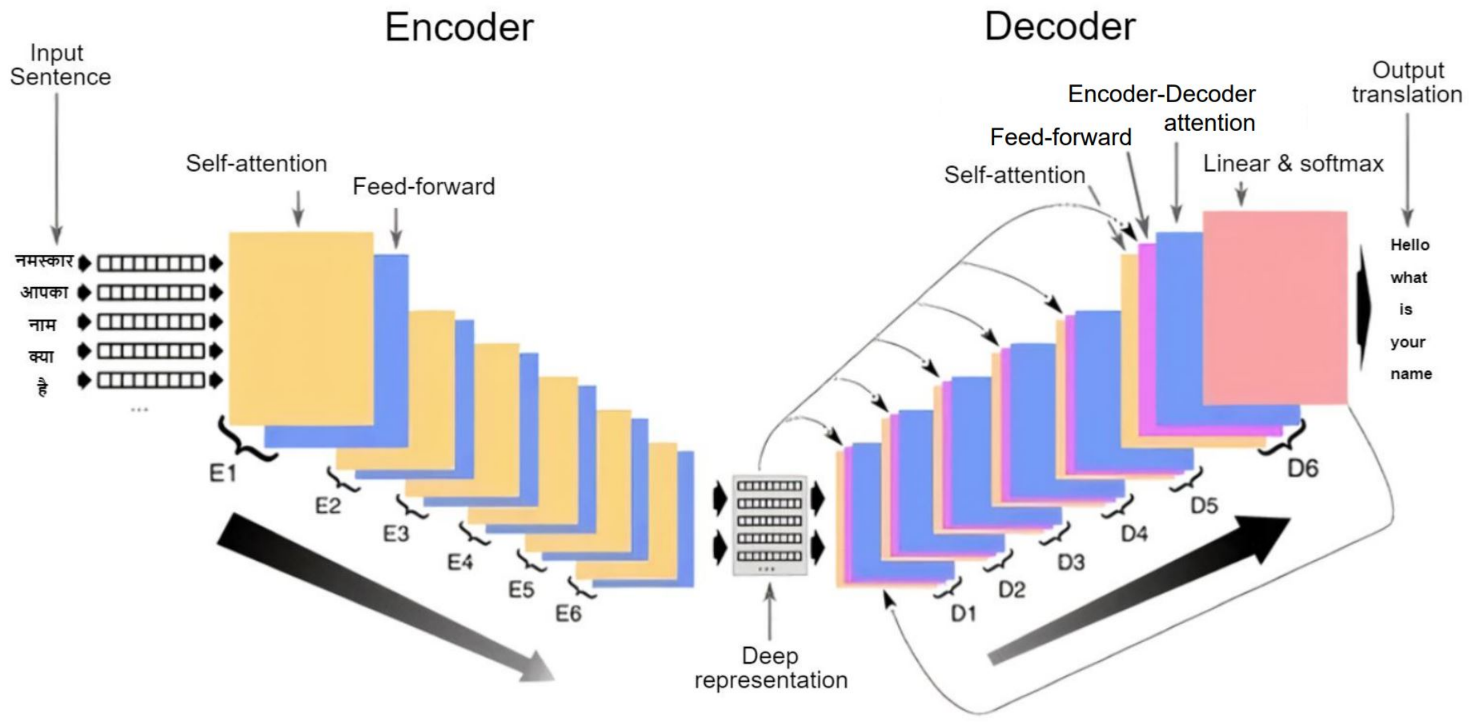


Figure 3.1: Model Architecture of the Machine translation system

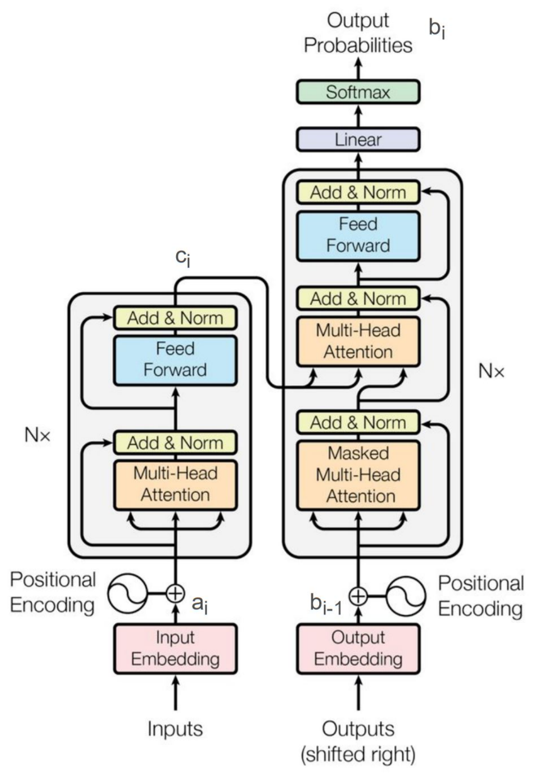


Figure 3.2: Transformer Sub layer architecture [29]

Now let us look at the pictorial representation of the architecture of a transformer as shown in Fig. 3.2. It consists of self attention layers shown in the left part of Fig. 3.2 and also point wise fully connected layers shown in the right part of Fig. 3.2. The left half of the transformer is used in each encoder layer as shown in Fig. 3.1 using two sub layers in each layer and the right half of the transformer is used in the each decoder layer as shown in Fig. 3.1 using three sub layers in each layer.

3.1 Encoder Stack and Decoder Stack

3.1.1 Encoder

The job of the encoder is to convert the input word tokens into an embedding format that can be further used by the decoder while providing the translation. As seen in Fig. 3.1, 6 layers make up the encoder, where every layer is further divided into 2 sub-layers. The first of these sub-layers is a mechanism that uses self attention which is multi headed in nature, multi-headed attention will be covered in the section 3.2.2. The second layer is a feed forward network, which is fully

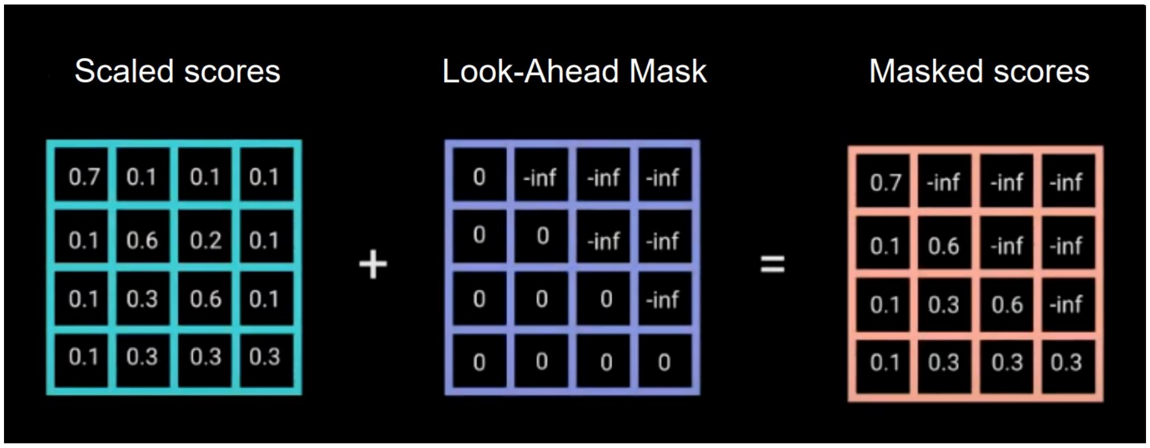


Figure 3.3: Look ahead mask

connected position wise. A residual connection [8] is used in both sub layers, the result of which is then subjected to layer normalization [1].

3.1.2 Decoder

The job of the decoder is to the take the embedded input from the encoder and convert them into the their corresponding translations and provide the word in the target language as the final output. As seen in Fig. 3.1 the decoder also consists of 6 layers, where each layer is further divided into 3 sub-layers. The initial two sub-layers are similar to those used in the encoder, the third sub-layer is used to perform encoder-decoder attention on the output generated by the encoder. As in the case of the encoder, a residual connection is used here and this output undergoes layer normalization. But there is a subtle modification to the self-attending sub-layer in the decoder, the combination of the look-ahead mask, as shown in Fig. 3.3, and the offset of output embeddings by one position makes sure, that the prediction for the symbol in position i depends only on the previously generated outputs.

3.2 Attention

When a sentence is composed in any language, there are words within it that have some interrelationship. To capture this relationship between words within a sentence, the concept known as "attention" was formed. It decides which word in the sentence pays how much attention to another word in that sentence. Attention [29] may be understood as a relationship between queries and a set of pairs, whose nature is that of a key and value, to the output. Here, query(q), key(k) and value(v) are all vectors. For each value, a weight will be assigned. This weight is computed by using a similarity function of a query with its respective key. Now the output can be calculated as a weighted sum of these values. There are two kinds of attention that are used in the machine translation system, and they are discussed below.

3.2.1 Attention using Dot Product

For calculating attention, we need an input that is made up of queries q and keys k which have dimensions of d_k. We also need values v which has a dimension of d_v. Queries in machine translation model can be understood as the input word vector, keys are the input word vectors for all the other words and values are the collection of positional input embedded vectors which are shown in the flowchart in Fig. 3.4. The mathematical representation of attention is given as follows :

$$AT(q,k,v) = S(qk^T/\sqrt{d_k})v \tag{3.1}$$

Where AT is attention, S is the softmax activation function and q,k,v are queries, keys and values, respectively. These attention vectors are computed for each word. Upon computation, they will contain the information regarding which word is being paid the most attention, by the encoded word.

3.2.2 Multi-Headed Attention

By using mutli-headed attention [29] we allow the model to simultaneously attend to various representation sub-spaces at various locations, instead of using a sole attention function. The mathematical representation of multi-headed attention is given as:

$$M(q,k,v) = [head_1, ..., head_h]A^o \tag{3.2}$$

Where M is a multi headed attention computed for a specific set of query, key, and value vectors. Here [] represents concatenation operation of the multiple heads

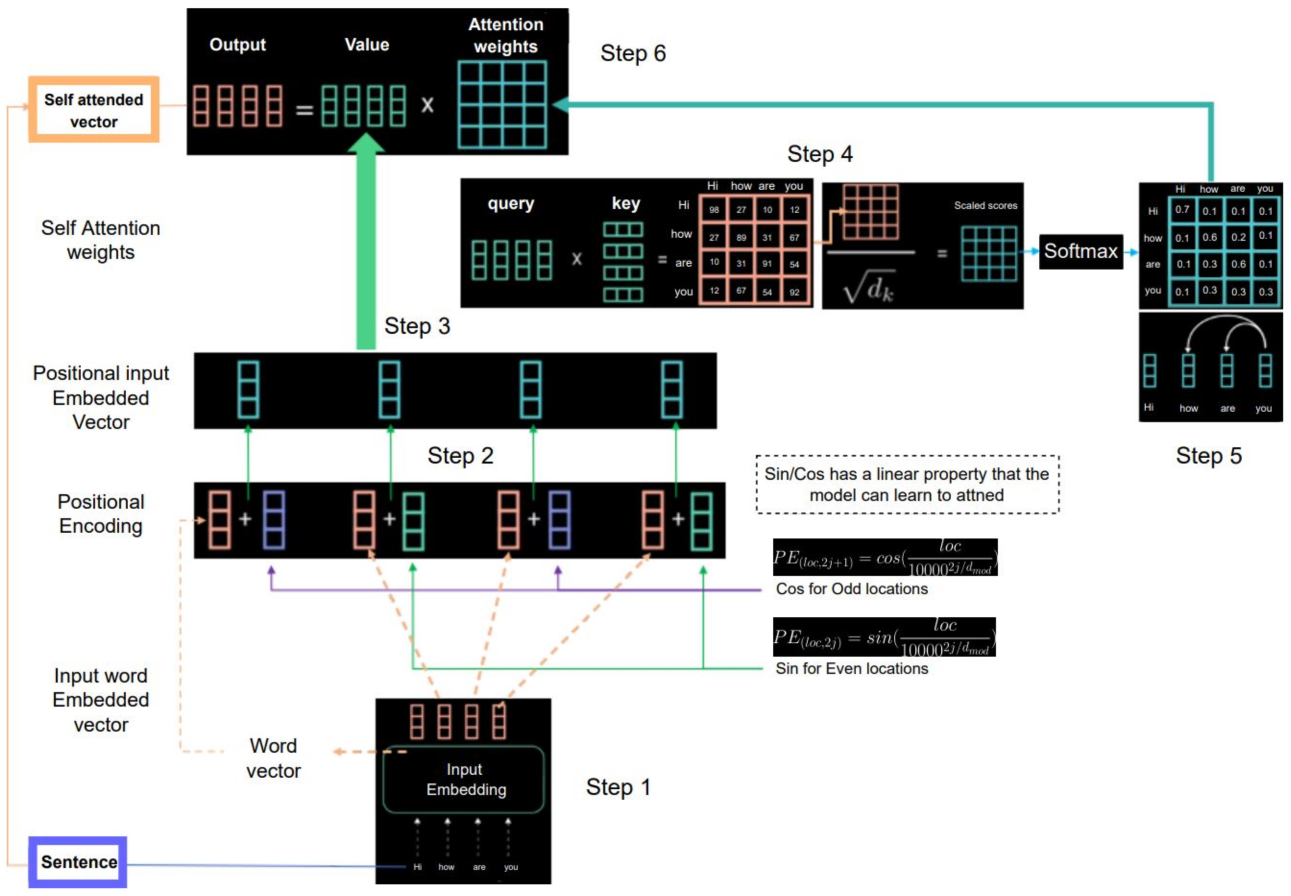


Figure 3.4: Process for forming self attention vector

taken from 1 to h, where the value of h is taken as 8 as per [29]. It is necessary to take multiple heads as it increases the model's ability to focus on various positions. It also prevents the word from dominating the encoding when an attention vector is calculated.

Each $head_i$ is computed as:

$$head_i = AT(qA_i^q, kA_i^k, vA_i^v) \tag{3.3}$$

The projections A_i^q, A_i^k, A_i^v, and A^o used above are parameter matrices, these parameters will be learned as the model is trained on the desired data and will be adjusted accordingly. Their dimensions are given as follows:

$$A_i^q \in \mathbb{R}^{d_{mod} \times d_k}, A_i^k \in \mathbb{R}^{d_{mod} \times d_k}, A_i^v \in \mathbb{R}^{d_{mod} \times d_v}, A^o \in \mathbb{R}^{hd_v \times d_{mod}} \tag{3.4}$$

Here q, k and v are projected h times linearly while using various linear projections, that can be learned. The dimensions d_k and d_v are set to 64 and d_{mod} is set to 512 according to [29]. When the attention function is utilized parallelly on each of the versions of q, k and v that are projected, they will generate output values of dimension d_v. These outputs are concatenated and then are projected once more, which then give us the final attention vector of each word.

3.3 Usage of Attention in Machine Translation

In Transformer, attention is used in different ways:

- To enable each individual position within the encoder to pay attention to all the positions within the earlier layer, the layers utilize self attention within the encoder, in these layers the q, k and v all are taken from the output generated by the encoder in the previous layer.
- The decoder also contains self-attention layers, which enable each individual position of the decoder to pay attention to the positions in the decoder. If the flow of information is leftward, it will interfere with the auto regressive property. To prevent this, a look ahead mask as shown in Fig. 3.3 is utilized. It is used to pay attention till the current position, including the current position.
- In the encoder decoder attention layer, each individual position in the decoder is enabled to pay attention to each position in the input sequence. The q are taken from the output of the earlier decoder layer and the k and v are taken from the output generated by the encoder.

3.4 Feed Forward Networks

A fully connected layer accompanies each layer in the encoder as well as the decoder. It is applied identically and separately for each position. It is made up of

ReLU activation between two linear transformations. It is given by:

$$F(x) = max(0, xA_1 + c_1)A_2 + c_2 \tag{3.5}$$

3.5 Softmax and Embeddings

For converting the input tokens as well the output tokens into vectors which have a dimension of d_{mod}, embeddings are used. A softmax function is used in combination with the learned linear transformation to help predict the probabilities of the next word.

3.6 Position wise Encoding

This translation model does not contain any operations that are recurrent or convolutional in nature. This creates a need to include the information regarding the relative or absolute positions of the words in the sequence, which will help the model to detect the arrangement of the sequence. This is where positional encodings are used. The inputs are first embedded and then the positional encodings are added to it in the encoder stack as well as the decoder stack. The dimensions of both these encodings are the same d_{mod}. For this purpose, sine function as well as cosine function with contrasting frequencies are used. They are given below.

$$PE_{(loc,2j)} = sin(\frac{loc}{10000^{2j/d_{mod}}}) \tag{3.6}$$

The above equation uses the sinusoidal function to keep track of the words occurring at even locations in a sentence and the below equation uses the cosine function to keep track of the words occurring at odd locations in a sentence.

$$PE_{(loc,2j+1)} = cos(\frac{loc}{10000^{2j/d_{mod}}}) \tag{3.7}$$

Here loc is the location of the vector and j is the dimension of the vector as can be seen in Fig. 3.4. The positional encoding vector will follow a particular pattern. The model learns this pattern through which it will help establish the location of each word or the separation between different words in the sentence. These positional encoding vectors are added to every input encoding. By doing this, the encodings are provided with relevant separation of the encoding vectors after they are projected into the query, key, and value vectors for the computation of the attention vector.

3.7 Algorithm

In traditional sequence to sequence models that are made up of an encoder decoder structure using RNN or LSTM, the encoder will process the sequence given in the input into a context vector of constant length. This context vector is fed to the decoder as an input. Using this, the decoder starts predicting the output. But the problem associated with a fixed length context vector is that it is unable to remember sequences which are longer. It tends to forget the beginning part of the sequence after the whole sequence is processed. This is the motivation for the development of the attention mechanism. For the purpose of understanding how the attention mechanism works, let us walk through an example where a Hindi sentence is converted into an English sentence. The algorithm is explained next.

3.7.1 Computation of score for Encoder State

Each encoder state E_1, E_2, E_3, E_4 and E_5 helps in the storage of local information of the sequence given as input. The objective is to predict the first word of the output translation, but the decoder is not provided with any initial state. So the output of the last encoder state E_5 is used as the input to the first decoder state. A feed forward network is now trained using every encoder state and also the present decoder state. Let us say the information to predict the first word in the translated output is stored in the encoder states E_1 and E_2. Therefore we need the decoder to pay more attention these states rather than the others. This is the reason for training a feed forward network so that it will learn to allocate a higher score to the states which require more attention and allocate a lower score to the states that have less significance. Let us consider that S_1, S_2, S_3, S_4 and S_5 are the generated scores for each encoder state by the feed forward network, which will be used later on to compute the attention weights. Since the information is in E_1 and E_2 their scores S_1 and S_2 will be higher compared to the others.

3.7.2 Computation of the Attention Weights

A softmax is applied on the scores that are generated to produce attention weights w_1, w_2, w_3, w_4 and w_5. In this particular example the values of w_1 and w_2 will be higher compared to the others to help predict the first word.

3.7.3 Computation of the Attention Vector

After the generation of the attention weights, an attention vector will be generated that can be utilized by the decoder. The decoder is now able to predict the next word that occurs in the sequence. The attention vector AV is calculated as:

$$AV = \sum_{i=1}^{5} w_i * E_i \tag{3.8}$$

In this case, the values of w_1 and w_2 are higher compared to the others, therefore it contains more information from the states E_1 and E_2.

3.7.4 Decoder output

The decoder makes use of this attention vector and the word that was generated in the earlier time step to predict the upcoming word that occurs in the sequence. Since in the initial time step, there is no output generated from the earlier time step, a special token $START$ is given to the decoder. The decoder predicts the first word of the sequence and also generates a hidden state d_1.

- **Decoding at time step 2**: To predict the upcoming word in the sequence, the internal state d_1 along with all the encoder states E_1, E_2, E_3, E_4 and E_5 are given to the feed forward network which then generates new S_1, S_2, S_3, S_4 and S_5 scores which are used to compute new attention weights and a new context vector is generated. This attention vector is compounded with the output of the earlier time step and is given to the decoder to predict the next word also producing a new internal state d_2.

This process is sustained till the END token is generated by the decoder. After the generation of this token the process is terminated. It is to be noted is that in the case of traditional Seq2Seq models, a fixed context vector is used for every decoder time step, whereas in the case of attention mechanism, a new attention vector is computed every time by using newly generated attention weights.

3.8 Methodology

When working with any model that is based on the transformer architecture, the requirement for training data is huge. As a result, the training time, as well as the resources required, are quite high as can be seen from the English to French translation implemented in [29], which took 3.5 training days on eight NVIDIA P100 GPUs on a dataset of 36M English-French sentence pairs. For this reason, experiments are usually performed on a pre-trained model. For the purpose of this work, a pre-trained multilingual model opus-mt was chosen. This model supports the translation of nearly 127 Indo-European languages into English.

Initially the performance of this model, based on transformer architecture, is evaluated against a fraction of two custom made data sets. These custom data sets will be described in chapter 4. A baseline score is generated on the pre-trained model for each data set. After this, the base model is fine-tuned on the remaining fraction of the two custom data sets in increasing steps of data size. The fraction, which was initially used to evaluate the base model was used again to evaluate the newly fine-tuned models.

The model used here is established on a transformer-based NMT. This model utilises Marian-NMT framework. This framework is a NMT toolbox which is production-ready and stable. It offers efficient training and decoding potential. The architecture is based on a standard transformer setup utilizing 6 self-attentive layers. Both the encoder as well as the decoder make use of 6 layers. The network makes use of 8 attention heads h in each layer.

Since the purpose of this model is to support translation among multiple languages and not translate the sentences where the input consists of multilingual components, it will have to be modified so as to perform the task at hand, and therefore this model will be subjected to fine-tuning on the custom data sets. We will now look into fine-tuning in more detail in the following section.

3.8.1 Fine-tuning

"Fine-tuning" refers to the process of taking a pre-trained model and training it again on the desired custom data sets. The process of fine-tuning adjusts and updates the weights of the pre-trained model so that it can accommodate the traits of the custom data set and perform the desired task. Fine-tuning a pre-trained model improves its ability to generalize. The process of fine-tuning is pictorially represented in the Fig. 3.5. The process of fine-tuning involves the following steps:

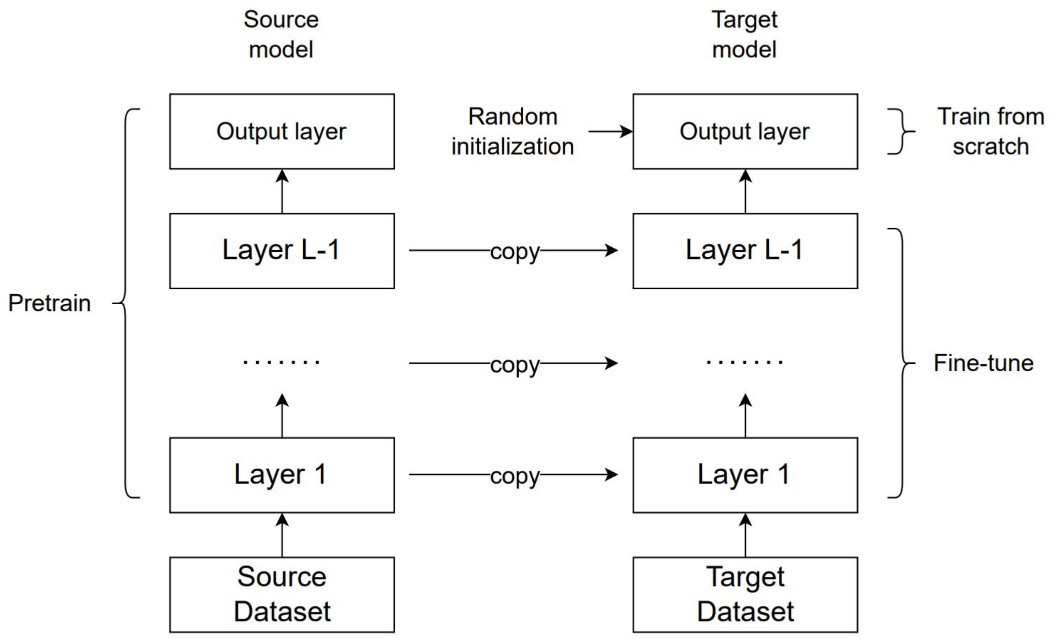


Figure 3.5: Fine-tuning a model

- Initially a pre-trained model can be picked, or a model can be trained to perform the required base operations. In this case the base operation would be the translation of a Hindi sentence to an English sentence.
- Now a target model is created which contains all the model designs and parameters of the source model excluding the layer from where the output is generated. It is assumed that the parameters of the model contain the information learned from the source dataset. This information may be relevant to the target dataset. It is also assumed that the labels of the source dataset are closely related to the output layer of the source model, therefore the output layer is not used in the new target model.
- A new output layer is added to the target model, where the number of outputs is equal to the number of classes in the target dataset. Then the parameters of this new output layer are randomly initialized.
- Now the target model is trained on the custom dataset. During this process, the output layer is trained from the beginning and the parameters of the other layers are adjusted according to the parameters of the source model.

CHAPTER 4

Experiments and Results

4.1 Evaluation Metric

The ideal evaluation metric for evaluating the performance of machine translation is human evaluation, as it considers many aspects of a translation such as fidelity, adequacy, and fluency of the translation. It may be an extensive approach but it is very expensive as takes months for the evaluation process. This shows the need for an automatic evaluation metric of translation, which is quick, inexpensive, language-independent, and can correlate greatly with human evaluation.

4.1.1 Bleu Score

Bleu score [20] was created as a replacement for human evaluation. The idea behind bleu score is to determine the closeness of a machine translation to its corresponding human translation, the more similar the machine translation is to the human translation the better. Bleu score is computed as follows:

$$P = \frac{m}{w_t} \tag{4.1}$$

Where P is unigram precision, m is the number of words from the generated translation that are found in the reference translation and w_t is the total number of words in the generated translation. This quantity shows what percentage of words in the generated translation are matching with the reference translation.

$$p = \begin{cases} 1 & if\, c > r \\ e^{\left(1-\frac{r}{c}\right)} & ifc \leq r \end{cases} \tag{4.2}$$

Where p is the brevity penalty which gives a penalty with an exponential decay to the generated translations that are too short because they cannot be compared to the reference translations. Here c is the length of the generated translation and r is the length of the reference sentence.

$$BLEU = p \cdot e^{\sum_{n=1}^{N}\left(\frac{1}{N} * log P_n\right)} \tag{4.3}$$

Here N is the N-gram size. It takes into consideration how many words must be considered for each gram. The value of N is set to 4. This bleu score is used as an estimate of the model performance by evaluating the model generated translation against the reference translation.

4.2 Dataset

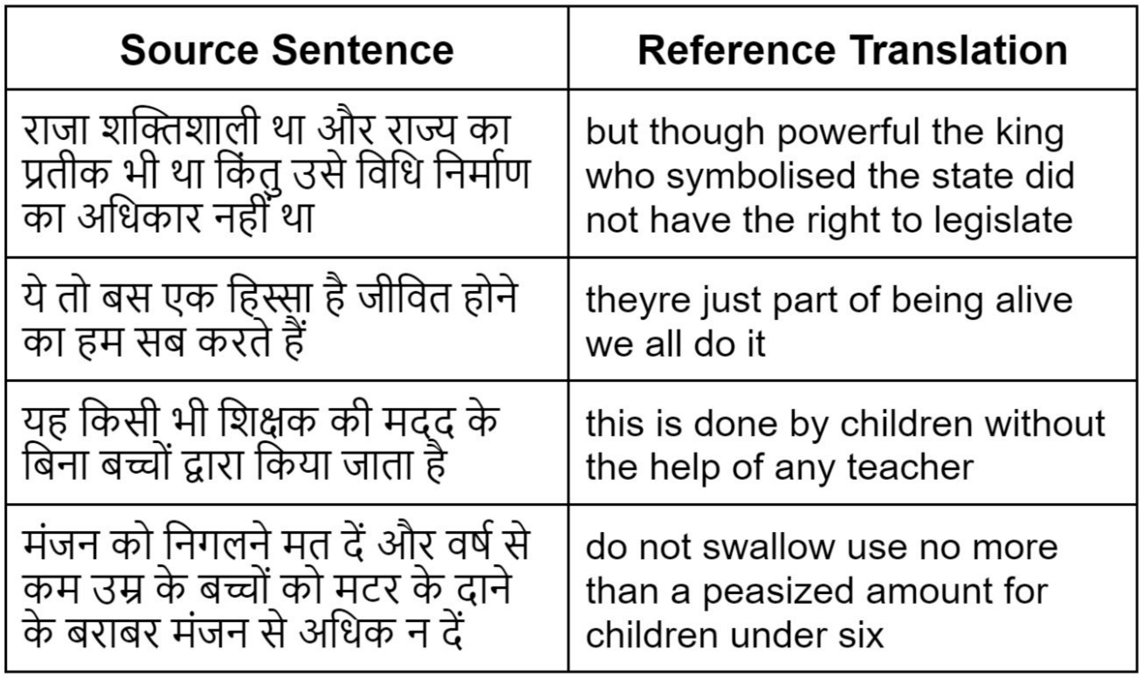
The model that was inferred was originally trained on the OPUS [28] data set which consists nearly 200 languages with over 3.2 billion sentences and fragments of sentences which contain 28 billion tokens. The model was trained to behave as a multilingual translation model with the capability of translating 59 languages into English.

Creation of data for machine translation is a difficult task as it requires a highly skilled professional to give proper translations of a given source sentence. For the scope of this work the custom data sets were created with the help of an already existing dataset. For the experiments mentioned in this work, two custom data sets were created. They are described as follows:

4.2.1 Data-1

The purpose of this dataset is to simulate the situation where a translation is required when the input is a sentence formed by the combination of two languages, where one half of the sentence is in one language and the other half in a different language.

This custom data set was created with the help of the sentences from the HindiEnCorp [3] dataset which consists of nearly 1,27,000 Hindi-English sentence pairs. The Hindi-English sentence pairs provided by the HindiEncorp dataset can be seen in Fig. 4.1. The custom data set consists of three variations of sentence pairs such as a) A single sentence in Hindi with its corresponding English translation, b) A sentence which begins in Hindi and ends in English with its combined translation in English and c) A sentence which begins with English and ends with Hindi with its combined translation in English. The variations b) and c) were created by using two consecutive sentence pairs from the HindiEnCorp dataset. This custom data set consists of 95,642 such sentence pairs. The sentence pairs of this data set can be visualized as shown in Fig. 4.2.



Source Sentence	Reference Translation
राजा शक्तिशाली था और राज्य का प्रतीक भी था किंतु उसे विधि निर्माण का अधिकार नहीं था	but though powerful the king who symbolised the state did not have the right to legislate
ये तो बस एक हिस्सा है जीवित होने का हम सब करते हैं	theyre just part of being alive we all do it
यह किसी भी शिक्षक की मदद के बिना बच्चों द्वारा किया जाता है	this is done by children without the help of any teacher
मंजन को निगलने मत दें और वर्ष से कम उम्र के बच्चों को मटर के दाने के बराबर मंजन से अधिक न दें	do not swallow use no more than a peasized amount for children under six

Figure 4.1: Illustration of the sentence pairs from HindiEncorp dataset

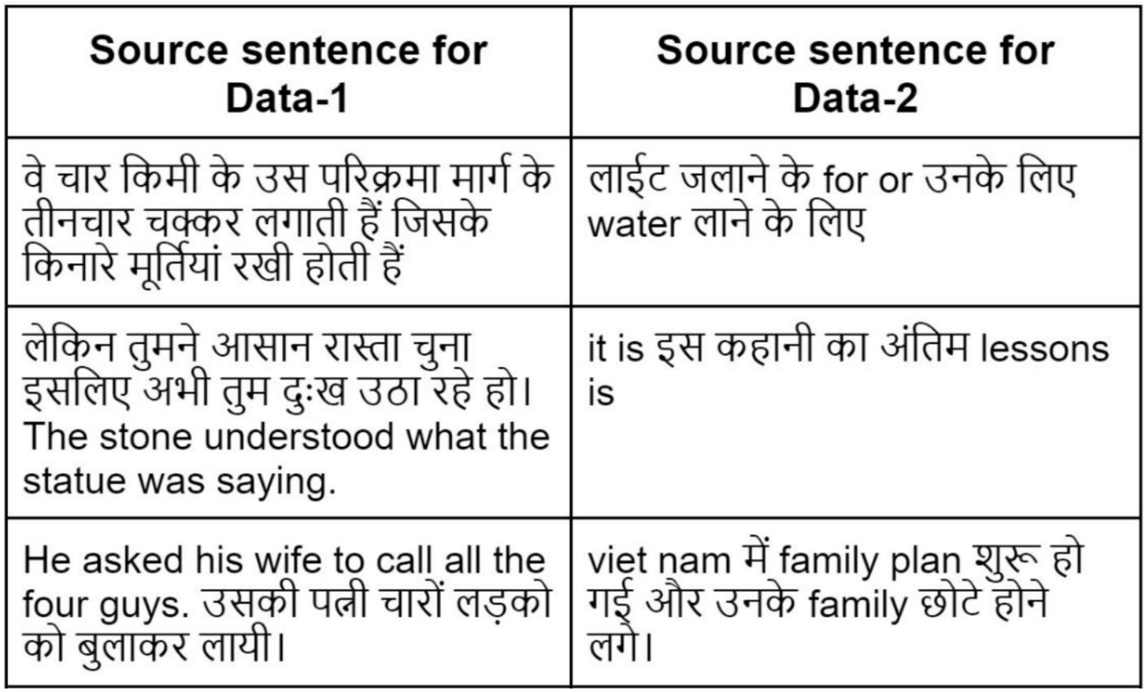
4.2.2 Data-2

The purpose of this dataset is to simulate the situation where a translation is required when the input is a sentence which is formed with words from two different languages and where the order in which the languages are used is not specific as in the above case.

This custom data set was also created with the help of the sentences from the HindiEnCorp dataset. For its creation, first a Hindi sentence was taken and a random number was generated in the range of 1 to half of the sentence length. This is the number of words that will be replaced from the sentence in English. Only half of length was used so as to not completely replace the sentence. After this random indices are generated, total indices equal to the random number generated previously, the word at this random index within the sentence is replaced with its English counterpart. 10,000 such pairs were created. The sentence pairs of this data set can be visualized as shown in Fig. 4.2.

4.3 Training and Testing

We have two custom data sets, to evaluate the performance of the model on them, experiments were carried out in the following manner:



Source sentence for Data-1	Source sentence for Data-2
वे चार किमी के उस परिक्रमा मार्ग के तीनचार चक्कर लगाती हैं जिसके किनारे मूर्तियां रखी होती हैं	लाईट जलाने के for or उनके लिए water लाने के लिए
लेकिन तुमने आसान रास्ता चुना इसलिए अभी तुम दुःख उठा रहे हो। The stone understood what the statue was saying.	it is इस कहानी का अंतिम lessons is
He asked his wife to call all the four guys. उसकी पत्नी चारों लड़को को बुलाकर लायी।	viet nam में family plan शुरू हो गई और उनके family छोटे होने लगे।

Figure 4.2: Illustration of the two custom data sets

4.3.1 For Data-1

The model was trained on a fraction of the custom data set. This fraction has been iteratively increased across various experiments and the results are shown in Table 4.1. The first 10,000 sentence pairs of the custom data set were fixed as the testing data set across all the experiments. Initially the sentence pairs in the range of 10k to 20k were selected as the training set and the performance of the model was evaluated after training it on this data set. This process was repeated while increasing the size of the training data set by 10,000 during each experiment and the performance was noted. For these experiments the number of epochs was set to 1, and a batch size of 1 were fixed. Adamw [17] was used as the optimizer here therefore a weight decay of 0.01 was fixed.

The effect of varying the weight decay from 0.01 to 0.1 over a fixed training set of sentence pairs in the range of 10k to 20k while the testing set is depicted in Table 4.2. When the number of epochs was increased from 1 to 2 while taking a weight decay of 0.01, and training the model using sentence pairs in the range of 10k to 20k, the model showed improvement in performance.

Training set	Testing set	Bleu Score
Pretrained	first 10k	18.333
10k-20k	first 10k	18.942
10k-30k	first 10k	18.481
10k-40k	first 10k	18.951
10k-50k	first 10k	19.112
10k-60k	first 10k	19.078
10k-70k	first 10k	19.079
10k-80k	first 10k	19.191
10k-90k	first 10k	19.017

Table 4.1: Bleu Score with varying size of training data for Data-1

4.3.2 For Data-2

The model was trained on a fraction of the custom data set. This fraction has been iteratively increased across various experiments and the results are shown in Table 4.3. The first 2,000 sentence pairs of the custom data set were fixed as the testing data set across all the experiments. Initially the sentence pairs in the range of 2k to 2.5k were selected as the training set and the performance of the model was evaluated after training it on this data set. This process was repeated while increasing the size of the training data set by 1,000 during each experiment and the performance was noted. For these experiments the number of epochs was set to 1, a batch size of 1 and weight decay of 0.01 were fixed.

4.4 Results for Data-1

When the base model was evaluated against the test set, a bleu score of 18.333 was observed. Upon fine-tuning the base model on the custom data set as shown in Table 4.1, increasing the size of the training set almost always improves the bleu score and always performs better than the base model. Consequently increasing the weight decay while keeping other parameters constant has shown an improvement in the bleu score as can be seen in Table 4.2. Furthermore when the number of epochs was increased from 1 to 2 while keeping the other parameters constant showed an improvement in the performance which produced the highest bleu score of 19.827, amongst all the experiments carried out so far.

Training set	Testing set	Weight decay	Bleu Score
10k-20k	first 10k	0.05	18.929
10k-20k	first 10k	0.07	18.931
10k-20k	first 10k	0.1	19.124

Table 4.2: Bleu Score with varying Weight Decay for Data-1

Training set	Testing set	Bleu Score
Pretrained	first 2k	8.884
2k-2.5k	first 2k	12.027
2k-3k	first 2k	18.348
2k-4k	first 2k	16.456
2k-5k	first 2k	17.668
2k-6k	first 2k	16.071
2k-7k	first 2k	15.242
2k-8k	first 2k	14.705
2k-9k	first 2k	14.228
2k-10k	first 2k	13.922

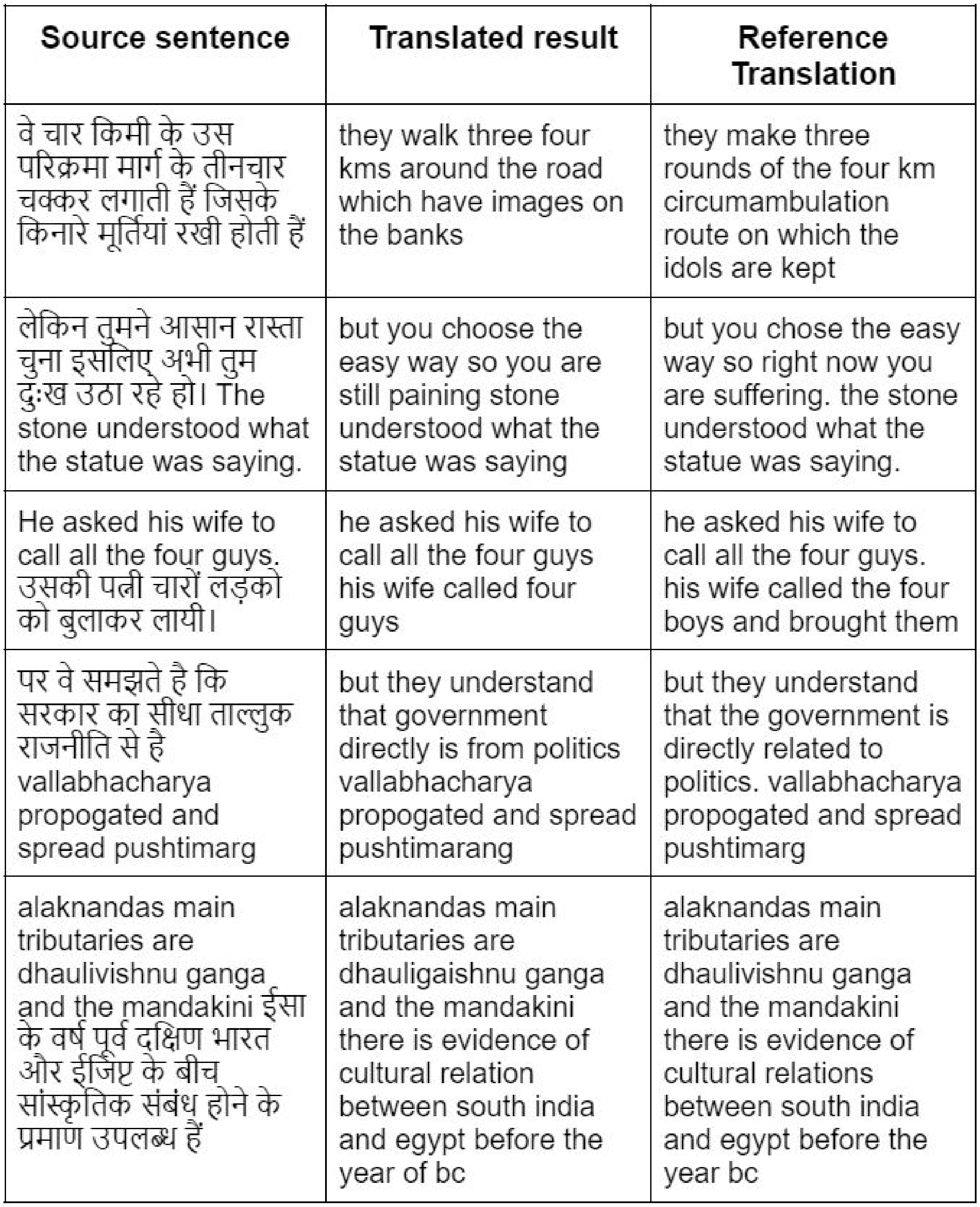
Table 4.3: Bleu Score with varying size of training data for Data-2

4.5 Results for Data-2

When the base model was evaluated against the test set, a bleu score of 8.884 was observed. Upon fine-tuning the base model on the custom data set as shown in Table 4.3, increasing the size of the training set after a certain point resulted in a decrease of performance of the model.

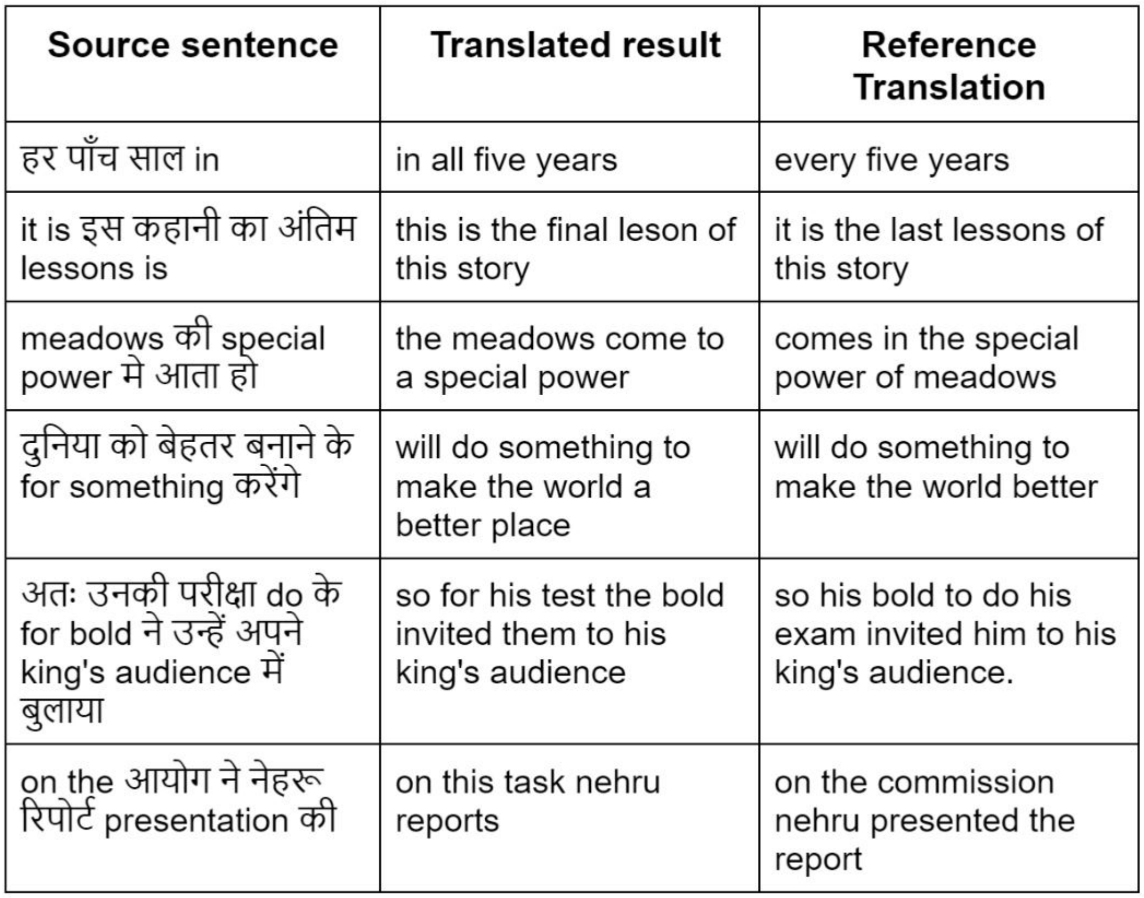
One should note that the method utilized for the creation of the custom data set 2 does not guarantee the preservation of context, due to the fact that random words were picked to be replaced. This can degrade context in cases where a combination of words in one language has a translation which has fewer number of words than the source phrase. So, by replacing even one word within that combination of words, its translation will be affected.

The generated results in both the cases are illustrated in Fig. 4.3 and Fig. 4.4. And the trends followed by the model upon varying the size of the training data in both the cases are shown in Fig. 4.5 and Fig. 4.6.



Source sentence	Translated result	Reference Translation
वे चार किमी के उस परिक्रमा मार्ग के तीनचार चक्कर लगाती हैं जिसके किनारे मूर्तियां रखी होती हैं	they walk three four kms around the road which have images on the banks	they make three rounds of the four km circumambulation route on which the idols are kept
लेकिन तुमने आसान रास्ता चुना इसलिए अभी तुम दुःख उठा रहे हो। The stone understood what the statue was saying.	but you choose the easy way so you are still paining stone understood what the statue was saying	but you chose the easy way so right now you are suffering. the stone understood what the statue was saying.
He asked his wife to call all the four guys. उसकी पत्नी चारों लड़को को बुलाकर लायी।	he asked his wife to call all the four guys his wife called four guys	he asked his wife to call all the four guys. his wife called the four boys and brought them
पर वे समझते है कि सरकार का सीधा ताल्लुक राजनीति से है vallabhacharya propogated and spread pushtimarg	but they understand that government directly is from politics vallabhacharya propogated and spread pushtimarang	but they understand that the government is directly related to politics. vallabhacharya propogated and spread pushtimarg
alaknandas main tributaries are dhaulivishnu ganga and the mandakini ईसा के वर्ष पूर्व दक्षिण भारत और ईजिप्ट के बीच सांस्कृतिक संबंध होने के प्रमाण उपलब्ध हैं	alaknandas main tributaries are dhauligaishnu ganga and the mandakini there is evidence of cultural relation between south india and egypt before the year of bc	alaknandas main tributaries are dhaulivishnu ganga and the mandakini there is evidence of cultural relations between south india and egypt before the year bc

Figure 4.3: Translation result for the model fine-tuned on Data-1



Source sentence	Translated result	Reference Translation
हर पाँच साल in	in all five years	every five years
it is इस कहानी का अंतिम lessons is	this is the final leson of this story	it is the last lessons of this story
meadows की special power मे आता हो	the meadows come to a special power	comes in the special power of meadows
दुनिया को बेहतर बनाने के for something करेंगे	will do something to make the world a better place	will do something to make the world better
अतः उनकी परीक्षा do के for bold ने उन्हें अपने king's audience में बुलाया	so for his test the bold invited them to his king's audience	so his bold to do his exam invited him to his king's audience.
on the आयोग ने नेहरू रिपोर्ट presentation की	on this task nehru reports	on the commission nehru presented the report

Figure 4.4: Translation result for the model fine-tuned on Data-2

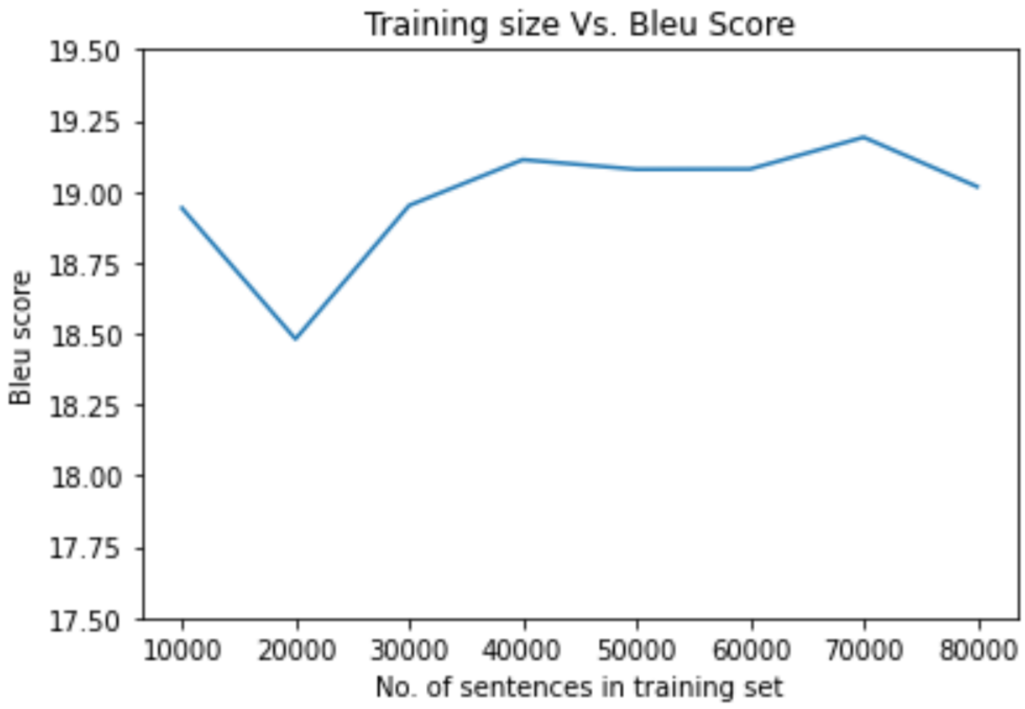


Figure 4.5: Result of varying the size of training data set for Data-1

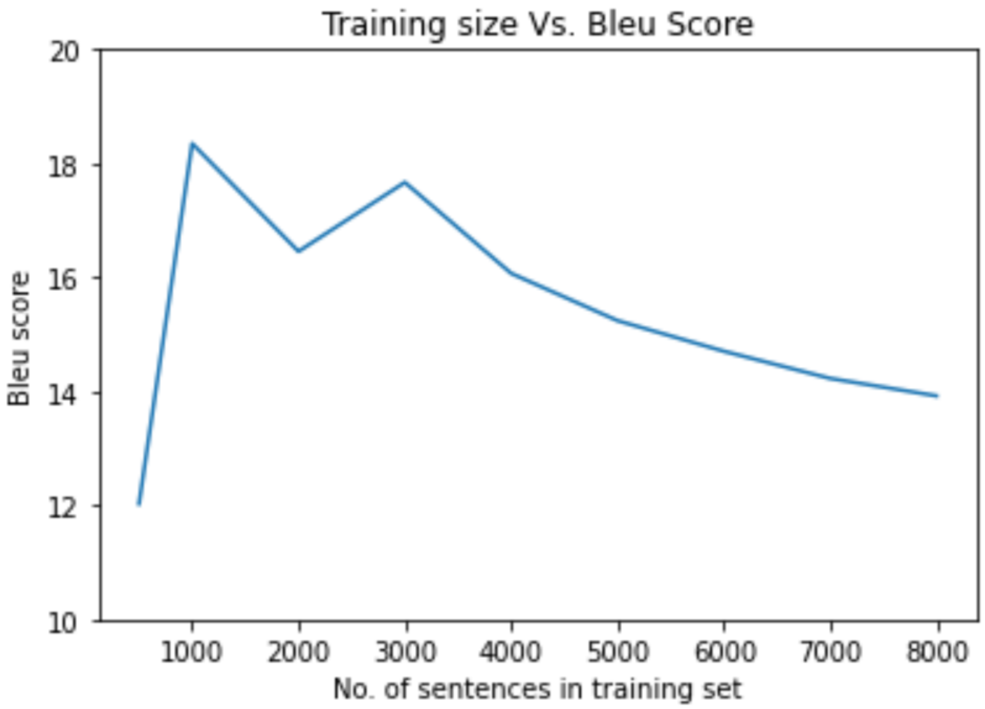


Figure 4.6: Result of varying the size of training data set for Data-2

CHAPTER 5

Conclusions and Future Work

In this work, the scope of fine-tuning the pre-trained multilingual model has been explored on two different kinds of custom data sets for the translation of multilingual sentences. For the scope of this work only the languages Hindi and English were used, but the same experiments can be carried out with different languages. The first dataset consists of source sentences in which one half of the sentence is in one language and the other half is in another language. The second dataset consists of source sentences in which some words have been replaced with their translations in another language. Utilizing these custom data sets, experiments have been carried out while fine-tuning the base model with varying size of the training data and the performance of the model was observed. It was observed that regardless of the size of training data, the performance of the fine-tuned model was better as compared to base model in both the custom data sets.

These results show that the transformer is capable of handling the translation of multilingual sentences. It translates these sentences with a decent bleu score, but the performance of this model can be improved if the model can be provided with a dataset where the multilingual sentences have more contextual relationship.

A data set with more contextual relationship can be created for multilingual sentences containing the languages Hindi and English by taking the Hinglish data set [26] and replacing the code-mixed hinglish text with its corresponding devanagari equivalent. This newly formed dataset can then be used to perform the experiments in this work to yield better results. Upon construction of datasets including other language pairs in the format specified in this work, work can be done to study the effect of a specific language pair on translation.

One possible application of the model established in this work is to develop a system with it as the base, that can work as a translator between two parties who communicate using a mixture of two or more languages but have one language in common.

References

[1] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] O. Bojar, V. Diatka, P. Rychlỳ, P. Straňák, V. Suchomel, A. Tamchyna, and D. Zeman. Hindencorp-hindi-english and hindi-only corpus for machine translation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3550–3555, 2014.

[4] J. Cheng, L. Dong, and M. Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

[5] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[6] M. R. Costa-Jussa and J. A. Fonollosa. Latest trends in hybrid machine translation and its applications. *Computer Speech & Language*, 32(1):3–10, 2015.

[7] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. Convolutional sequence to sequence learning. In *International Conference on Machine Learning*, pages 1243–1252. PMLR, 2017.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[9] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[10] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[11] R. Jozefowicz, O. Vinyals, M. Schuster, N. Shazeer, and Y. Wu. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*, 2016.

[12] Ł. Kaiser and S. Bengio. Can active memory replace attention? *Advances in Neural Information Processing Systems*, 29, 2016.

[13] N. Kalchbrenner, L. Espeholt, K. Simonyan, A. v. d. Oord, A. Graves, and K. Kavukcuoglu. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099*, 2016.

[14] A. Lavie, S. Vogel, L. Levin, E. Peterson, K. Probst, A. F. Llitjos, R. Reynolds, J. Carbonell, and R. Cohen. Experiments with a hindi-to-english transfer-based mt system under a miserly data scenario. *ACM Transactions on Asian Language Information Processing (TALIP)*, 2(2):143–163, 2003.

[15] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017.

[16] A. Lopez. Statistical machine translation. *ACM Computing Surveys (CSUR)*, 40(3):1–49, 2008.

[17] I. Loshchilov and F. Hutter. Fixing weight decay regularization in adam. 2018.

[18] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[19] L. Medsker and L. C. Jain. *Recurrent Neural Networks: Design and Applications*. CRC press, 1999.

[20] 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 of the Association for Computational Linguistics*, pages 311–318, 2002.

[21] A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*, 2016.

[22] R. Paulus, C. Xiong, and R. Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017.

[23] R. Rapp and C. Martín-Vide. Example-based machine translation using a dictionary of word pairs. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, 2006.

[24] F. Saleh, W. Buntine, G. Haffari, and L. Du. Multilingual neural machine translation: Can linguistic hierarchies help? *arXiv preprint arXiv:2110.07816*, 2021.

[25] S. Sreelekha. Statistical vs rule based machine translation; a case study on indian language perspective. *arXiv preprint arXiv:1708.04559*, 2017.

[26] V. Srivastava and M. Singh. Hinge: A dataset for generation and evaluation of code-mixed hinglish text. *arXiv preprint arXiv:2107.03760*, 2021.

[27] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 27, 2014.

[28] J. Tiedemann. Opus–parallel corpora for everyone. In *Proceedings of the 19th Annual Conference of the European Association for Machine Translation: Projects/Products*, 2016.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.