Advances in Nonlinear Variational Inequalities
ISSN: 1092-910X
Vol 28 No. 3s (2025)

Transformer based Neural Network Architecturefor Regional Language Translation

[1*]Deepak Mane, [2]Geetanjali Sharma, [3]Abhijit Banubakode, [4]Veena M. Kadam, [5]Sandip Shinde, [6]Sunil Sangve,

[1,5,6]Vishwakarma Institute of Technology, Bibwewadi, Pune-411037, Maharashtra, India

[2]Pimpri Chinchwad College of Engineering, Pune- 411044, Maharashtra, India

[3]MET Institute of Computer Science, Mumbai- 400050, Maharashtra, India

[4]CRG solutions Pvt Ltd., Mumbai-400097, Mumbai Maharashtra, India

[1*]dtmane@gmail.com, [2]geetu2k3@gmail.com, [3]abhijitsiu@gmail.com, [4]kadamveena1234@gmail.com, [5]sandeep.shinde@vit.edu, [6]sunilsangve@gmail.com

Correspondence author: dtmane@gmail.com

Article History:

Received: *30-09-2024*

Revised: *21-11-2024*

Accepted: *30-11-2024*

Abstract:

Today, the importance of regional language translation has grown significantly as it facilitates effective communication, accessibility, and inclusivity for diverse populations in a globalized world. It enables individuals to access information and services in their native languages, fosters cultural preservation, and enhances opportunities for education and economic growth. So, this paper presents a thorough implementation of the Transformer-based neural network architecture for regional language translation. Our methodology enables translation between languages by utilizing cross-attention and the self-attention mechanism. Because the architecture is modular in nature, it is simple to configure model parameters such as dropout rates, model dimensions, and layer count. We present feed-forward layers, layer normalization tokenization, positional encoding, and multi-head attention components. The Transformer model, when used in both the encoder and the decoder, may acquire intricate linguistic connections and patterns. We showcase the model's potential for natural language processing tasks by demonstrating its effectiveness in translating text between language with the help of Samanantar Dataset, a parallel corpus containing a total of 4,000,000 English and Kannada sentence pairs. The results showed a consistent rise in training accuracy with each epoch, highlighting the model's capacity to learn and generalize from the data, with notably enhanced performance correlated with larger datasets and more potent hardware utilization. This work offers a flexible foundation for future research in the subject and advances the creation of cutting-edge machine translation models.

Keywords: transformer, translation, regional languages, encoder, decoder, attention mechanism, natural language processing

Introduction

The demand for efficient language translation has never been greater in our era of growing globalization. Since language obstacles frequently obstruct communication and the exchange of ideas, artificial intelligence researchers are placing a high premium on the development of effective translation

techniques. The automatic translation of text between languages, known as machine translation, has long been the focus of intensive study and development.

With a variety of techniques and procedures designed to increase translation accuracy and fluency, machine translation has developed over time [1]. Statistical techniques and rule-based systems were the mainstays of early methodologies, which frequently failed to capture the subtleties and complexity of many languages [2]. Although these techniques laid the groundwork for machine translation, they were unable to handle the wide range of languages.

The advent of neural networks has significantly changed machine translation. By learning and encoding language patterns, these models could pave the way for the creation of increasingly complex and context-aware translation systems. RNN and LSTM networks gained popularity [3]. They could provide better translations by modelling dependencies and sequences. They were still having trouble with context preservation and long-range dependencies, though.

Vaswani et al's transformer architecture which was released in 2017 has made a significant advancement in machine translation [4]. Transformers used a cutting-edge self-attention method that allowed the model to take into account a sentence's whole context, leading to an unprecedentedly effective capture of long-range relationships. The self-attention and multi-head attention mechanisms enable the model to focus on different parts of the input text, resulting in precise translations. This Transformer architecture quickly gained popularity, and models showed its effectiveness on various NLP applications including BERT, GPT, and T5.

Transformers offered a number of improvements over previous approaches. They were an adaptable option for cross-lingual translation since they could manage lengthy phrases, capture context, and adjust to different language patterns. Our paper presents the development of transformer based neural network for cross lingual translation, with an emphasis on translating text between Kannada and English and allowing users to configure model parameters. This approach provides a possible answer to the cross-lingual language translation problem by utilizing the strength of self-attention mechanisms to overcome the difficulties provided by different linguistic characteristics and idiomatic expressions in both languages.

The purpose of this paper is to significantly improve translation efficiency and accuracy by outperforming other neural machine translation techniques, such as classical RNNs. Additionally, the proposed transformer model focuses on offering a unique advantage by allowing simplified configuration of model parameters, such as dropout rates, model dimensions, and layer count for its users. This flexibility makes it a versatile tool that can be easily adapted to specific language pairs and data constraints, meeting the demand for high-quality, accessible, and efficient regional language translation.

This paper includes a literature review which explores the transformative impact of Transformer models, especially in Neural Machine Translation, showcasing their wide-ranging applications in various languages. This paper also explores the potential of the Transformer model in regional language translation. It offers an in-depth methodology, training insights, and findings demonstrating how the model can be customised and perform better on bigger datasets and more capable hardware. This paper concludes by emphasizing the transformative impact of self-attention and the broader applications of Transformer-based models in various domains, promising advancements in linguistic subtleties and real-world adaptability.

I. LITERATURE SURVEY

In NLP, the development and enhancement of Transformer models have emerged as a revolutionary technology. This literature review embarks on a journey into the transformative world of Transformer models, with a particular focus on transformer creation using encoder decoder architecture. By delving into the intricacies of Transformer technology, this review seeks to unravel the underlying principles, innovations, and advancements that have reshaped the landscape of language processing, offering new perspectives and possibilities for language-related tasks beyond translation. The field of the machine translations, which sits at the intersection of linguistics and computer science, has advanced over time. These developments have been fuelled by the efforts to facilitate communication, across different languages. These advancements have not enhanced translation accuracy but also led to the creation of advanced systems that can effectively translate a wide range of languages.

One such machine translation technique that has demonstrated significant improvements over conventional machine translation algorithms for translation of a given language to other is neural machine translation (NMT). The author's focus here is on using neural translation methodologies to translate into Indian languages. For the translation of Telugu into English and vice versa, a Seq2Seq NMT model with an encoder-decoder attention mechanism was proposed. [5]. For Neural Sequence Modelling an open source and a faster approach is openNMT. Here, the proposed architecture will translate and provide audio output in Indian language from real-time English audio input. It overcomes the limitations of traditional approaches including RBMT and SMT [6]. Since language obstructions are arising in primary healthcare due to the rise in international migration, NMT is being used to advance speech-to-speech translation through the development of several translation models and techniques. The model was evaluated using a number of parameters, such as Mean Reciprocal Rank, Perplexity, Fluency scores, Adequacy scores, Bleu Scores, Precision, and Overall ratings. [7].

The standard approach to resolving the problem of linguistic barriers has not been effective or favourable. An androidd application was built that runs without the internet. It uses firebase for language translation which process images, text, scans the text from the image and translates that text to the desired output language chosen by the user [8]. Additionally, a portable speech to speech technology was developed for seamless communication between Iraqi colloquial and English. which includes speech recognition, question mapping, concept translation, and text-to-speech synthesis.[9]. The tourists usually found themselves struggling with language barriers during their trips, highlighting the need for a language translator app, So in order to satisfy the needs of tourists speaking different languages in quick conversations, a user-centred design for a real-time cross-lingual translation system was provided [10].

The proposal of the Transformer, an novel network architecture that solely employs attention mechanisms and eliminates the need for RNN or CNN. With the Transformer, multi-headed self-attention takes the place of recurrent layers, which are frequently employed in encoder-decoder designs, enabling quicker training for translation tasks. This model outperformed even previously developed ensembles, achieving BLEU scores of 41.0 for translation of English to French and BLEU score of 28.4 for translation of English to German [4]. The Word-Piece model was developed to enhance the effectiveness of translation for limited words by generating sub word units specifically for Arabic ethnicities, in contrast to approaches including large vocabularies and back-off dictionaries. The Arabic language translation quality was improved by the usage of sub word units as well as the effect of different encoders, decoders and attention head count in self-attention mechanism [11]. In recognition of the Kurdish language, a different transformer based NMT model was created using vocabularies dictionary units that are identical to the entire dataset. The model was trained on a substantial corpus that was created by combining the KurdNet—the Kurdish wordnet—with the TED, Auta, Tanzil, and

other sizable parallel corpora of Kurdish-English. As a result the model indicated a high-quality translation with a BLEU score of 0.45, demonstrating its effectiveness in translating Kurdish texts [12].

For the low-resource languages of South Africa, neural machine translation (NMT) models have also been developed. Utilising based transformer NMT with self-attention, these models have attained novel outcomes across many languages. Two NMT models, JoeyNMT and NMT, are then compared, and the BLEU score is used to assess each model's performance. Leveraging Bible data in nine different languages as its training set, the LID attained 99% accuracy. [13]. One such transformer that eliminates the problem of lengthy computation and computing error is described, leading to the output with the least amount of computational error impact. This is achieved by changing the architectural structure and combining layer outputs. It acquires a BLEU score of 36.7 thanks to a hybrid input strategy that combines the outcome from each of the layers with the initial input data to compensate for the gradual loss of the information influence as the network becomes deeper [14]. Some reachers used machine learning based deep learning approach for translating the English to different Indian languages [15][16]. A sequential method for comprehensive voice to voice translation using a Transformer-based model called Transcoder is proposed which addresses the limitations of traditional cascade-based systems by integrating ASR, MT, and TTS into a single model. It can learn long context information and find related information from the entire sequence using self-attention [16]. A comprehensive study of transformer models and how they are applied in machine translation is conducted, with a focus on Indian regional languages. It serves as a guide for researchers having various opportunities available in Indian linguistics which help researchers choose appropriate models and directions for their research giving them insight into the most recent cutting-edge models created in NLP using transformers [17].

II. METHODOLOGY

In this paper, we present a technique to the modern deep learning architecture known as a transformer model. Our methodology seeks to offer a clear and thorough summary of the procedures needed to build and train the transformer model, as well as the essential preprocessing methods and hyper-parameter tuning strategies.

Based on the general architecture shown in Figure 1, the Transformer utilises point-wise, completely linked layers across the encoder and the decoder in addition to layered self-attention.

The Encoder Decoder is an important development in sequence-to-sequence learning, notably in machine translation and natural language comprehension. The encoder module efficiently evaluates the input sequence, collecting word associations and producing rich contextual representations by employing multi-head self-attention technique. Using both self-attention and cross-attention methods to pay attention to significant portions of the input and output sequences, the Decoder module simultaneously aids in the generation of the target sequence. The Encoder and Decoder work together to enable the model to handle sequential input effectively by integrating positional encoding and feed-forward networks. This enables the translation and production of text in a seamless manner across many languages and contexts. Our methodology clarifies the intricate functioning of the Encoder and Decoder within the Transformer architecture, providing a comprehensive understanding of their functions in facilitating the models ability to capture complex linguistic patterns and produce contextually coherent and accurate translations.

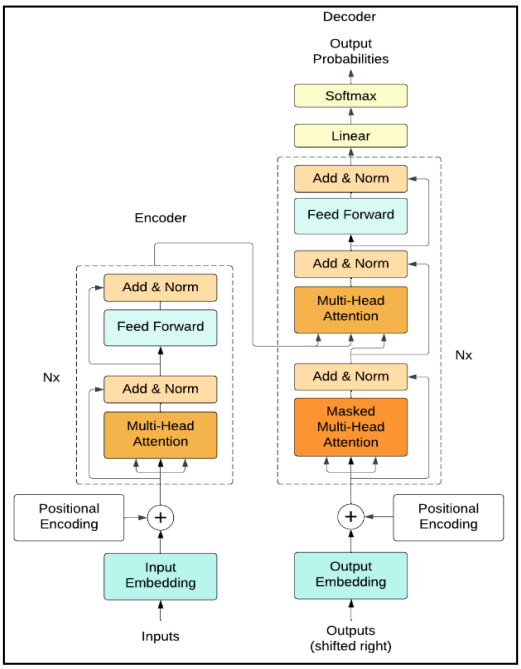


Figure 1. Transformer Architecture

A. Encoder

The encoder in the transformer architecture takes the input data and creates vector representations from the input sequence that include the necessary details. It comprises of six identical layers: multi-head self-attention, layer normalization and position wise feed forward networks.

i) Input Embedding

Here, the input text is given in English , which is the source language in our case. Word embedding creates a vector representation of every word in the phrase. These embeddings capture syntactical and semantic information that aids the model in comprehending word meaning within context. Input Embedding Steps includes:

1. Vocabulary and Tokenization: In this step, vocabulary is created from unique words in the training data. Then a sentence is tokenized into words and words are mapped to their vocabulary indices.

2. Embedding Layer: Then we use an embedding layer with a specified embedding dimension to map word indices to continuous vectors.

3. Word to Vector Mapping: For each word, we have to look up the corresponding vector in the embedding matrix and concatenate these vectors to create the input embedding for the sentence.

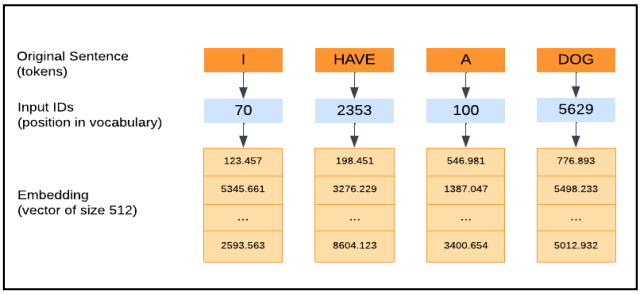


Figure 2. Input Embedding

We have used an embedding to convert the input words into continuous vector representations of size 512 as shown in Figure 2. This layer is learned during training and provides a dense and meaningful representation for each word in the sentence.

ii) Positional Encoding

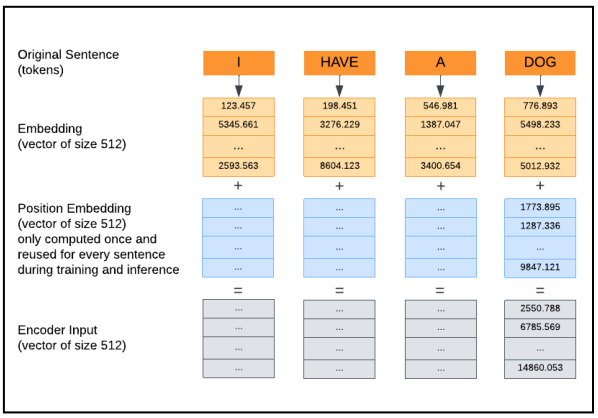


Figure 3. Positional Encoding

We employ positional encoding to overcome the Transformer architecture's fundamental limitation of knowledge regarding the order of words in a sentence.

Positional encoding adds position-specific information to the word embeddings. It informs model about the relative positions of words in the sentence thereby enhancing its ability to interpret word sequences.

We make use of a sinusoidal positional encoding mechanism, which efficiently captures positional information without introducing excessive complexity.

The formulae for the positional encoding are as follows:

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \quad (1)$$

$$PE(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \quad (2)$$

Where,

- **pos** specifies a word's position within the sequence.
- **i** is the positional encoding's dimension.
- $\mathbf{d_{model}}$ is the dimensionality of model
- **PE** represents the positional encoding.

The positional encoding is a vector with the identical dimension as that of input embeddings, providing each element with a unique positional encoding value.

iii) Multi-Head Attention

Multi-head attention is a crucial aspect of Transformer architecture that allows the model to focus on many input sentence portions simultaneously. This process makes it easier for the model to identify complex trends in the data and to gather different types of information.

In Multi-head attention we have multiple self-attention and we have used self-attention for the transformer's encoder and decoder. In the encoder, self-attention facilitates the model's reviewing of various sentence words during encoding. While the decoder helps the model to focus on different portions of the input text while creating the translation.

Here's an explanation of the methodology involved, along with some mathematical formulas:

1. Linear Transformation: Using a linear layer, the input sequence is first linearly converted into query, key, and value representations. This process involves projecting the input sequence into multiple subspaces to capture different aspects of the data.

2. Reshaping and Splitting: The resulting tensor from the linear transformation is reshaped to split the representations into multiple heads. This allows the model to perform parallel attention computations on different parts of the sequence. The splitting is typically done along the last dimension of the tensor.

3. Scaled Dot-Product Attention: The scaled dot-product attention is applied separately to each head.

It involves the following steps:

a. Calculating the attention scores between the query and key using the formula:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (3)$$

Where,

- **Q** represent the query vector.
- **K** represent the key vector.
- **V** represent the value vector.
- $\mathbf{d_k}$ denotes the dimension of the keys.

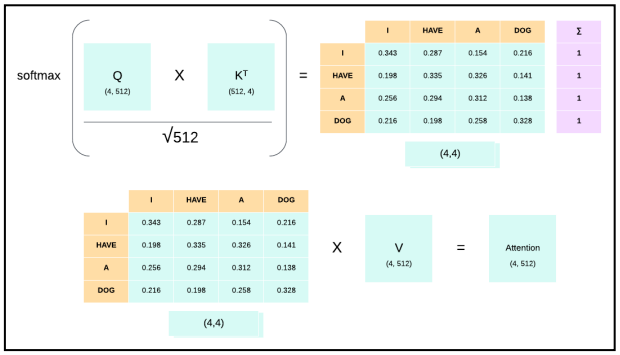
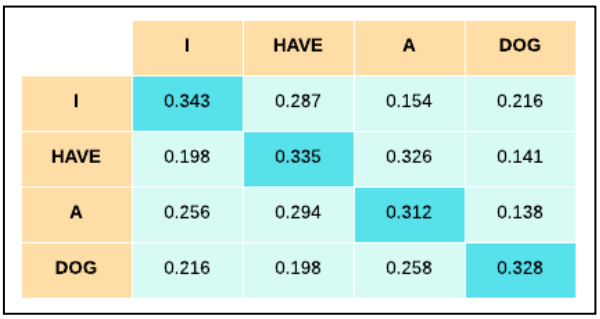


Figure 4. Computation of self-attention

Figure 4 demonstrates how self-attention is computed using above attention formula.



	I	HAVE	A	DOG
I	0.343	0.287	0.154	0.216
HAVE	0.198	0.335	0.326	0.141
A	0.256	0.294	0.312	0.138
DOG	0.216	0.198	0.258	0.328

Figure 5. Self-Attention

The dot-product values are scaled by the key dimension's square root to prevent the gradients from getting too small.

b. Applying the SoftMax function to obtain the attention distribution, which illustrates the relative significance of the various sequence components,

c. To generate the output, calculate the weighted sum of the data based on the attention scores.

4. Linear Projection and Concatenation: After the attention computations, the output from each head is linearly projected and concatenated to obtain a unified representation. This allows the model to obtain a comprehensive understanding of the input sequence, considering various aspects simultaneously.

Through the implementation of these steps, multi-head attention helps in capturing complex relationships and dependencies in the input text, enabling the model to learn and represent sequential information efficiently.

iv) Add and Norm

Following multi-head attention, the "Add and Normalize" method integrates data from different positions to stabilize the training process and enhance network convergence. It normalizes layer inputs to achieve standard normal distribution, thereby reducing the internal covariate shift. The steps involved in layer normalization are:

1. Computation of Mean and Variance: The mean and variance of the features are computed for every position in the input sequence. It helps in understanding the distribution of values for normalization.

2. Normalization: In order to help bring the activations to a standard scale, these input values are then normalized to have a value with a mean of 0 and a standard deviation (SD) of 1.

3. Scaling and Shifting: These normalized inputs then undergo scaling by a learnable parameter (gamma) and shifting by another learnable parameter (beta). These position-specific parameters maintain the network's representational capacity allowing the model to adapt and modify the normalized values uniquely for each position within the input sequence. During the training process, these parameters are then updated and fine-tuned, enabling the network to learn the most effective scaling and shifting for improved performance.

v) Position-wise feed-forward Network

The position-wise feed-forward network processes every position in the sequence independently, which causes the model to become non-linear. This network consists of two linear layers, which essentially perform linear transformations on the data. Between these layers, there is a ReLU (Rectified Linear Unit) activation function, which enables the model to represent complex and nonlinear relationships observed within the data.

Let X be the input tensor of shape (batch_size, sequence_length, d_model), where dmodel stands for the model's dimensionality. The Position-wise feed-forward networks function as follows:

$$FFN(X) = \max(0, XW_1 + b_1)W_2 + b2 \quad (4)$$

Here, W1, W2, b1, and b2 are learnable parameters in the form of weights and biases associated with linear transformation, and function max(0,x) is ReLU activation function which calculates element-wise maximum of 0 and the result of the linear transformation (XW1 + b1). This non-linearity network is used to identify complex relationships and trends in the data.

B. Decoder

Decoder is an important component in transformer which creates the output sequence using the previously created tokens and the encoded input.

Decoder starts with the conversion of the input sequence into embeddings with positional encodings to understand content and structure. The auto-regressive feature is preserved by the Decoder by employing a masked multi-head self-attention technique to ensure that the model only looks at prior positions during decoding. Furthermore, the Multi-Head Cross Attention facilitates communication between the Encoder and Decoder for contextually relevant translations. Layer normalization and residual connections stabilize training, while position-wise feed-forward networks capture complex patterns.

i) Input Embedding and Positional Encoding

An input sequence is then sent to the decoder, which converts it into a set of embeddings that correspond to each token or element in the sequence. It also provides positional encodings to give information about the order or position of the sequence's components.

ii) Masked Multi-Head Self-Attention Mechanism

As seen in Figure 6, the Decoder makes use of a masked multi-head self-attention mechanism to make sure that the model pays attention only to positions before the current position during the decoding process. This prevents the model from peeking ahead and ensures that the generated output adheres to the auto-regressive property.

	I	HAVE	A	DOG
I	0.343	0.287	0.154	0.216
HAVE	0.198	0.335	0.326	0.141
A	0.256	0.294	0.312	0.138
DOG	0.216	0.198	0.258	0.328

Figure 6. Masked Multi-Head Self-Attention

iii) Multi-Head Cross-Attention Mechanism

The Multi-Head Cross Attention approach let the encoder and decoder communicate with each other throughout the translation process. It makes it possible for the Decoder to focus on various segments of the encoded input sequence, which helps the model provide translations that are pertinent and correct. It consists of several steps

1. Key-Value Transformation from the Encoder: Linear transformations are applied to the encoded input sequence to project it into key and value representations. These transformations help in extracting meaningful information from the input sequence.

2. Query Transformation from the Decoder: In parallel, the Decoder generates query representations that capture the current state of the decoding process. These query representations are then transformed to facilitate effective comparison with the key representations from the Encoder.

3. Scaled Dot-Product Attention: It is used to compute the attention scores between the query and the key vector. Through this process, the Decoder can focus on important segments of the encoded input sequence that are necessary for generating appropriate translations

4. Weighted Sum and Linear Projection: It computes a weighted sum of the scores according to the attention scores so that the Decoder may access the most relevant data from the encoded input. The result is then linearly projected to ensure compatibility with the subsequent layers in the Decoder.

The Transformer model, which is a potent architecture for a variety of sequence-to-sequence tasks such as text summarization, machine translation and other cross lingual tasks, can effectively leverage the encoded input information to generate accurate and contextually relevant translations by integrating the Multi-Head Cross Attention.

iv) Layer Normalization and Residual Connections

In addition to the Encoder, the Decoder also employs layer normalization to enhance information flow and stabilize training. Afterwards, residual connections enable the original information to be maintained and sent via the layer from both the self-attention mechanism's input and output.

v) Position-wise Feed-Forward Networks

After that, the outputs are processed by the Decoder using position-wise feed-forward networks, which enables the model to identify complex patterns and relationships in the data and generating accurate output sequences

C. Algorithm

1. Import necessary libraries and modules
2. Define the Transformer architecture
3. Implement the Transformer architecture, including the Encoder and Decoder layers
4. Define custom loss and metric functions
5. Load and preprocess the dataset (including tokenization and padding)
6. Instantiate the Transformer model
7. Assemble the model with an optimizer, loss function, and metrics
8. Using the entire dataset to train the model.
9. Evaluate the model on the same dataset or new data
10. Make predictions with the trained model
11. Save the model weights and architecture for future use
12. Implement functions for translation

III. RESULTS AND DISCUSSION

This section presents the outcomes of our experiments, encompassing metrics such as accuracy and loss observed throughout the training of the Transformer-based model designed for English to Kannada translation. We also present our model's performance and analyze implications of the results.

Dataset:

For training and evaluation, we utilized the Samanantar dataset. Samanantar is the biggest publicly accessible parallel corpus collection for 11 Indian languages, encompassing Bengali, Assamese, Hindi, Kannada, Gujarati, Malayalam, Marathi, Oriya, Punjabi, Tamil, and Telugu. 49.6 million sentence pairs from English to Indian languages are present in the corpus. We used a parallel corpus containing

English and Kannada sentence pairs from Samanantar. This extensive dataset encompasses a total of 4,000,000 sentence pairs designed for training purposes. Our machine translation model has a strong foundation due to these pairs, which each includes English as a source sentence and its matching Kannada translation.

Hardware Setup:

The research was conducted using Python, with the PyTorch library as the backend for machine learning operations, and the GPUs accelerated the training of custom Transformer model. The custom-configured Transformer model, optimized for natural language processing tasks, was trained on a locally hosted environment using a workstation with Nvidia GTX 1650Ti. In addition to our locally hosted setup, we also harnessed the power of a remote GPU, the Tesla V100, to further augment our computational capabilities for model training and evaluation.

Hyper-parameter Setting:

Hyper-parameter settings are vital in shaping the behavior of our Transformer model when training it for English to Kannada translation. These settings are not learned during the training process rather they dictate the model's architecture, training dynamics, and determine key aspects such as the number of layers, attention mechanisms, learning rates, and batch sizes. This section describes the hyper-parameter settings employed in our English to Kannada translation model that we are training using this Transformer. The table below provides a brief overview of the hyper-parameter settings used throughout our experiments, shedding light on the fundamental selections that form the basis of our research.

Hyper-parameter	**Values**
Model dimension	512
Feed-Forward Network hidden units	2048
No of Attention heads	8
No of Encoder-Decoder layers	3
Dropout probability	0.1
Maximum sequence length	200
Batch size	30
Epochs	5
Loss function	CrossEntropyLoss
Learning rate	0.0001
Optimizer	Adam
Device	CUDA (GPU) if available, otherwise CPU

Table 1. Hyper-parameter Setting

Training:

We monitored how our Transformer model learned and improved throughout the training process. We used two different GPUs: Nvidia GTX 1650Ti and powerful Tesla V100. This dual-GPU approach helped us thoroughly in evaluating the model's performance and adaptability across diverse hardware environments.

It includes

i) Nvidia GTX 1650Ti:

Firstly, we have used GTX 1650 Ti GPU for training our Transformer model but with a reduced dataset size. While the Samanantar dataset comprised a total of 4,000,000 sentence pairs, we selectively used 500,000 sentence pairs for our training. Despite this scaled-down dataset, the Nvidia GTX 1650 Ti

GPU allowed us to perform training efficiently. The training process with this configuration took approximately 16 hours to complete.

ii) Tesla V100 GPU:

We make use of the Tesla V100 GPU's amazing capabilities for performing the second set of experiments. This decision was driven to evaluate how well the model could perform and handle larger dataset when using the best hardware setup available. Compared to the Nvidia GTX 1650 Ti, the Tesla V100 demonstrated significantly faster training time thereby reducing the time required to complete the training process. The use of this GPU allowed us to explore the model's performance with a larger dataset and deeper architectures effectively. Training the model on the Tesla V100 required approximately 5 hours, offering compelling evidence of the benefits of using high-end GPUs for machine translation tasks.

The training and validation accuracy and loss curves for the corresponding hardware setups and dataset sizes are shown in Figure 7. The x-axis displays the number of epochs, while the y-axis shows the accuracy and loss values.

In our research, the training accuracy grew consistently throughout each epoch, indicating that the model was able to acquire knowledge from and generalize from the training data. We observed improvement in accuracy with the use of more powerful hardware and the increase in the dataset size.

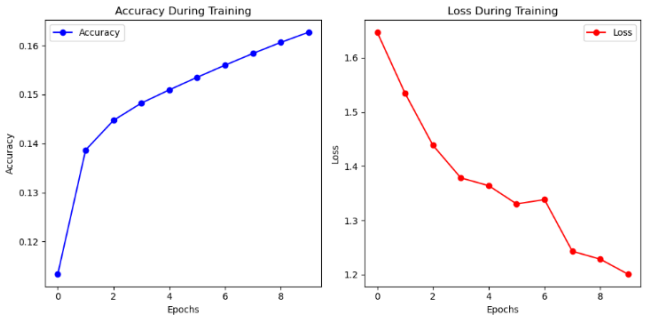


Figure 7. Learning accuracy & Loss graph

Performance Parameters:

From the Samanantar Dataset, on 1000000 sentence pairs for English-to-Kannada translation task with 10 epochs, the maximum BLEU score achieved by the Transformer model is 0.41.

```
English Sentence: A case has been registered at the city police station.
Actual Kannada Translation: ಎಂ. ಬಿ. ನಗರ ಪೊಲೀಸ್ ಠಾಣೆಯಲ್ಲಿ ಪ್ರಕರಣ ದಾಖಲಾಗಿದೆ.
Predicted Kannada Translation: ಘಂಬ ಎ. ಪಡರ ಪೊಲೀಸ್ ಠಾಣೆಯಲ್ಲಿ ಪ್ರಕರಣ ದಾಖಲಾಗಿದೆ.
English Translation of Predicted Kannada Translation:
(Ghamba A. A case has been registered at Padara police station.)
Bleu Score: 0.41
```

```
English Sentence:Its a part of our lives.
Actual Kannada Translation: ಇದು ನಮ್ಮ ಜೀವನದ ಭಾಗವೇ ಆಗಿದೆ.
Predicted Kannada Translation: ಇದು ನಮ್ಮ ಜೀವನದ ಪಾಗ.ಂ ನಗಿದೆ.
English Translation of Predicted Kannada Translation:
(This is a part of our life.)
Bleu Score: 0.26
```

Figure 8. Sample Outputs

We train the system on 100 epochs since introducing the MT module at higher values will help guarantee a good BLEU score. Neural networks are usually trained over several We train the system on 100 epochs since introducing the MT module at higher values will help guarantee a good BLEU score. Neural networks are usually trained over several Above figure shows sample output of our transformer model when trained for English-to-Kannada translation for two sentences

which includes actual English sentence with its actual Kannada translation, predicted Kannada translation and its English meaning with BLEU score.

We train the system on 100 epochs since introducing the MT module at higher values will help guarantee a good BLEU score. Neural networks are usually trained over several We train the system on 100 epochs since introducing the MT module at higher values will help guarantee a good BLEU score. Neural networks are usually trained over several

Comparative analysis of various machine translation methods:

Features	Rule-Based MT (RBMT)	Statistical MT (SMT)	OpenNMT (Seq2Seq)	Transformer
Notable Features	Rule-based, linguistic rules	Phrase-based or syntax based	Open-source, supports various NMT architectures	Self-attention mechanism, multi-head attention
Flexibility	Limited to predefined rules	Limited flexibility in handling complex language structures	Customizable, supports various configurations	Highly flexible, supports various tasks and languages
Quality of Translation	Highly dependent on rule quality and language coverage	Dependent on the quality of training data and language pairs	Quality depends on the data and model quality	State-of-the-art performance, high-quality translations with context awareness
Computational Complexity	Low	Medium to high	Varies depending on model and hardware	Varies depending on model and hardware
Benefits	Well-defined rules, deterministic approach, specific domains	Effective for various language pairs and domains, data driven translation	Customization, supports multiple languages, versatile for different NMT tasks	High-quality translations, context awareness, versatile for various NLP tasks, also suitable for low resource languages
Limitations	Limited to specific domains, not suitable for low resource languages	Requires large parallel corpora	Data-intensive, resource intensive, fine tuning may be required	Computationally intensive

Table 2. Comparative analysis of various machine translation methods

The table above presents a comprehensive comparison of various machine translation methods and architectures, including Rule-Based Machine Translation, Statistical Machine Translation, OpenNMT, and Transformer. Upon careful examination, it becomes evident that the Transformer architecture stands out as the most promising and versatile approach for translation tasks. With self-attention, multi head attention mechanism and context-awareness, the Transformer excels in delivering cutting-edge

performance and high-quality translations, making it the preferred choice for translation and other natural language processing applications.

IV. Conclusion and Future Scope

In this paper, we have presented a brief overview of Transformer-based neural network architecture for regional language translation with an emphasis on Samanantar dataset for English to Kannada translation. We have discussed various components of Transformer architecture, in encoder and decoder modules including feed-forward layers, positional encoding, layer normalization, tokenization and multi-head attention. Additionally cross-attention and self-attention mechanisms, enabling user-friendly configuration of hyper-parameters in the Transformer. The findings demonstrate the increase in model's training accuracy consistently improving across epochs, particularly with larger datasets and more potent hardware. This work demonstrates the transformational influence of Transformer-based models in addressing language intricacies and real-world flexibility, and it offers a flexible platform for future research.

The future work will be focus on expansion of language pairs for translation which will enable a broader reach across regional languages. Additionally, the model's performance can be enhanced by experimenting with deeper architectures and larger training datasets. Leveraging advancements in hardware, like more powerful GPUs and distributed computing, can lead to even more efficient training and deployment of these transformer-based models. Furthermore, the Transformer model may be used to various NLP tasks such as sentimental analysis, summarization of text, and question-answering systems.

References

[1] *Brown, P., & Johnson, M. (2007). Advances in Machine Translation. Machine Translation Journal, 14(2), 115-128.*

[2] *Chen, Y., & Wang, L. (2005). Statistical Machine Translation: A Comprehensive Review. Journal of Computational Linguistics, 33(2), 123-167.*

[3] *Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735-1780.*

[4] *Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I., 2017. Attention is all you need. Advances in neural information processing systems, 30.*

[5] *Jayanthi, N., Lakshmi, A., Raju, C.S.K. and Swathi, B., 2020, December. Dual translation of international and Indian regional language using recent machine translation. In 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS) (pp. 682-686). IEEE.*

[6] *Vyas, R., Joshi, K., Sutar, H. and Nagarhalli, T.P., 2020, March. Real time machine translation system for english to indian language. In 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS) (pp. 838-842). IEEE.*

[7] *Kushal HU, Kethu Yashaswi, Patil Chanchal Vinod, Meghana G,"Speech to Speech Translation with Response Suggestion (English-Hindi),"International Journal for Research in Applied Science & Engineering Technology, 2022*

[8] *M Vaishnavi, HR Dhanush Datta, Varsha Vemuri, L Jahnavi,"Language Translator Application,"International Journal for Research in Applied Science & Engineering Technology, 2022*

[9] *R. Prasad et al., "Real-Time Speech-to-Speech Translation for PDAs," 2007 IEEE International Conference on Portable Information Devices, Orlando, FL, USA, 2007*

[10] *Pires, Eduardo & Vital, Lais & Alves, Carina & Gomes, Alex. (2011). Voice interfaces for real-time translation of common tourist conversation. 232-236.*

[11] *Baniata, Laith H., Isaac. K. E. Ampomah, and Seyoung Park. 2021. "A Transformer-Based Neural Machine Translation Model for Arabic Dialects That Utilizes Subword Units" Sensors 21, no. 19: 6509.*

[12] *Badawi, Soran. (2023). Transformer-Based Neural Network Machine Translation Model for the Kurdish Sorani Dialect. UHD Journal of Science and Technology. 7. 15-21. 10.21928/uhdjst.v7n1y2023.pp15-21.*

[13] *Sefara, T.J., Zwane, S.G., Gama, N., Sibisi, H., Senoamadi, P.N. and Marivate, V., 2021, March. Transformer-based machine translation for low-resourced languages embedded with language identification. In 2021 conference on information communications technology and society (ICTAS) (pp. 127-132). IEEE.*

[14] *Abrishami, M., Rashti, M.J. and Naderan, M., 2020, April. Machine Translation Using Improved Attention-based Transformer with Hybrid Input. In 2020 6th International Conference on Web Research (ICWR) (pp. 52-57). IEEE.*

[15] *Upadhye, G. D., Kulkarni, U. V., & Mane, D. T. (2021). Improved Model Configuration Strategies for Kannada Handwritten Numeral Recognition. Image Analysis and Stereology, 40(3), 181-191. https://doi.org/10.5566/ias.2586*

[16] *Mane, D., Patil, M., Chaudhari, V., Nayakavadi, R., Pandhe, S. (2023). A Survey on Chatbot in Devanagari Language. In: Pati, B., Panigrahi, C.R., Mohapatra, P., Li, KC. (eds) Proceedings of the 6th International Conference on Advance Computing and Intelligent Engineering. Lecture Notes in Networks and Systems, vol 428. Springer, Singapore. https://doi.org/10.1007/978-981-19-2225-1_31*

[17] *Kano, T., Sakti, S. and Nakamura, S., 2021, January. Transformer-based direct speech-to-speech translation with transcoder. In 2021 IEEE Spoken Language Technology Workshop (SLT) (pp. 958-965). IEEE.*

[18] *Dixit, R., 2023. A Comprehensive Review of Transformer models and their Implementation in Machine Translation specifically on Indian Regional Languages. Available at SSRN 4449023.*