JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Language Translation using Efficient pretrained NLP Models

Harikrishnan. S
Department of Information Technology
SRM University, SRM Institute of Science and Technology, Kattankulathur-603203.

Abstract – Machine Translation is one of the NLP task, model that translates the given input text source language to another target language using Deep Learning models and it follows the sequence of process steps text preprocessing, Linguistic Analysis, word Embeddings, Language Modelling and Translate text Generation. The advancement of Language Translation has been significantly driven by Artificial Intelligent Deep Learning language models. In this paper aimed to translate the text from English to German Language and do the enhancement of existing application and evaluate the performance, Language Translation developed by using Deep Learning neural network models, transformers based models and Autoregressive based models. This paper explores the design and working functionality of a Language Translation based on intelligent pretrained NLP models and highlighting the key components used in models , methodologies employed in modern systems. These systems utilize NLP pre-trained models like LSTM and seq-seq models encoder and decoder, attention mechanism, Transformer Models which are fine-tuned for specific language pairs to achieve high-quality translations. The paper discusses the role of NLP and techniques, including neural machine translation (NMT), and encoder-decoder architectures, attention mechanisms, Transformer Architecture (generates human like output results) by enhancing the accuracy, fluency, and contextual understanding of translations. Furthermore, we address the challenges associated with Language translation, low-resource languages, and the computational demands of these models. Finally, the paper explores enhancement of existing system and developed multilanguage modal translation systems using LLMs based Google Translator(internally using NMT and attention mechanism, multi head Transformer architecture), as well as the potential for zero-shot examples. and analyze more about performance through an examinate , Evaluate the base model and Multi Language Model Language Translation performance by using BLEU score. this paper provides insights into the capabilities and future of Language Translation systems based on intelligent language models.

Keywords - Language Translation, Machine Translation MT, NMT Neural Machine Translation, seq-seq encoder decoder, Attention Mechanism, Transformer Architecture. LLM, Multi Language Translator. BLEU.

I. Introduction

Language translation is one of the most widely used applications of Natural Language Processing (NLP), enabling the automatic conversion of text or speech from one language to another. The primary goal of a language translation system is to break down language barriers, facilitating communication between people who speak different languages.

Language Translation within the Natural Language Processing (NLP) that uses computer algorithms to automatically translate text from one language to another, essentially allowing machines to understand and convert human language across different language speakers without manual intervention; this is performed through techniques NLU Natural Language Understanding and Natural Language Generation, its like analyzing syntax, semantics, and

context within the source text to produce an accurate translation into the target language

Machine Translation NLP systems have evolved from simple rule-based systems to sophisticated models Statistical machine translation (SMT) and Neural machine translation (NMT) Utilizing deep learning neural networks to learn complex language patterns and generate translations with higher accuracy more accurate and making the process faster and context-aware. NLP models to understand, interpret, and generate human like language in a way that is both meaningful and useful. The goal is to take a input text in one language (source language) and generate an equivalent text in another language (target language) using computational methods.

Language Machine Translation methods

Machine Translation (MT) refers to the use of computational models to automatically translate text or speech from one language to another. various methods and techniques have evolved to improve the quality and accuracy of translations

Rule-Based Machine Translation (RBMT)

This method relies on a set of pre-defined linguistic rules, dictionaries, and grammar rules for both the source and target languages. The system analyzes the sentence structure in both languages and applies transformations based on rules.

Components:
- Syntax Analysis: Breaks down sentences into grammatical structures.
- Lexical Transfer: Maps individual words to their equivalents in the target language.
- Syntactic Transfer: Alters sentence structure to match that of the target language.

Advantages:
- High precision in specific language pairs.
- Transparent and interpretable rules.

Challenges:
- Hard to scale for large numbers of languages.
- Requires extensive manual effort to develop rules and dictionaries.
- Not great with handling idiomatic expressions or ambiguity.

Statistical Machine Translation (SMT)

SMT uses statistical models to translate text based on the likelihood of word or phrase alignments between the source and target languages. It leverages large bilingual corpora to learn translation patterns.

Techniques:
- Word-Based Translation Models: Translates text at the word level.
- Phrase-Based Models: Groups words into phrases and translates entire phrases instead of individual words.

Advantages:

- Better than rule-based methods for handling large corpora.
- Works well when large parallel corpora are available.

Challenges:

- Struggles with long-range dependencies or context.
- Translation quality often drops with complex sentence structures or idioms.

Neural Machine Translation (NMT)

NMT uses deep learning models, particularly Recurrent Neural Networks (RNNs, Long Short-Term Memory (LSTM) networks, and Transformers to model the entire translation process as a sequence-to-sequence problem. The model is trained end-to-end, learning how to map an input sequence (source language) to an output sequence (target language).

Encoder-Decoder Architecture: An encoder processes the input sentence, and a decoder generates the translated sentence.

Attention Mechanisms: These help the model focus on different parts of the sentence during translation and brings into the context, making NMT especially good at handling long sentences and complex structures.

Advantages:

- Produces fluent, high quality translations.
- Can handle complex sentence structures and longer sentences more effectively than previous methods.

Disadvantages:

- Requires large amounts of data and computational resources for training.
- More difficult to explain or interpret compared to rule-based methods.

II. Existing System

Language-specific Europe parliment conversations English to German translation NMT used Deep Learning models LSTM is a type of RNN, encoder-decoder based on the sequence-to-sequence model. We denote the encoder and the decoder language-specific scenarios, both are considered independent modules that can be freely interchanged to work in all translation directions. This system relies on pre-defined linguistic rules and dictionaries to translate between languages. It uses syntactic and grammatical rules to convert the sentence structure of the source language (English) into the target language (German). Primarily sentence-based, limited context awareness. Limited to text-based systems, no real-time translation in spoken conversations

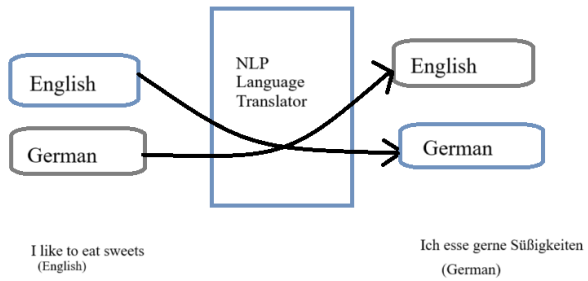

Fig. 1 – Language Translator

NMT systems process entire sentences takes as sequences, considering context across the sentence rather than translating word by word.

Challenges in existing system Struggles with idiomatic expressions, slang, and complex sentence structures. Requires large datasets and computational power, and there may still be errors with complex grammar or specialized terms.

Dataset

Europal dataset is used for language translation, Europe parliament conversations are translated into English, German languages. for translation there are limited parallel corpora

Translation Process Sequence

- Data input
- Data preprocessing
- Build Model LSTM
- Sequential() API
- Train the data
- Seq – Seq Encoder – Decoder
- Test Data
- Generate Output
- Performance Evaluation
- Results

Sentences from each dataset are cut into smaller sentences to get a smaller training set which has English sentences and German sentences. This now is used to train the model. This is done as to limit the size of the dataset and ensure it can work on a machine with a limited computing capability. The different datasets are combined into respective English and German datasets to provide a consolidated dataset. We first read these files into lists to clean the data and perform tokenization. In general preprocessing, we make all the data in lower case, remove the spaces at front and back of the dataset and remove the punctuation marks. Further pre-processing is required to make sure that each index of the list contains a sentence. In the second pre-processing, we tokenize all the words of the sentence and get a tokenized sentence and a tokenizer.

After preprocessing data is combined to build our final parallel dataset. The dataset used in encoded UTF-8 format.

```
english_vocab = len(eng_tokenizer.word_index) + 1
german_vocab = len(ger_tokenizer.word_index) + 1

print("English vocabulary is of {} unique
words".format(english_vocab))
```

```
print("german vocabulary is of {} unique
words".format(german_vocab))


MetaData of Dataset

Dataset : Europal
English vocabulary is of 13215 unique words
german vocabulary is of 22207 unique words


# Add this line to define english_vocab
english_vocab = len(eng_tokenizer.word_index) + 1
german_vocab = len(ger_tokenizer.word_index) + 1

# and add this line to define maxlen_eng
maxlen_eng = int(len(max(eng_tokenized, key=len)))

print("Maximum Length of English tokens is
{}".format(maxlen_eng))
print("Maximum Length of German tokens is
{}".format(maxlen_ger))

print('english_vocab', english_vocab)
print('german_vocab', german_vocab)

Maximum Length of English tokens is 222
Maximum Length of German tokens is 136
english_vocab 12911
german_vocab 22133
```

III. Architecture

The architecture and key components for an English-to-German language translation system using an LSTM Encoder-Decoder model.

Language translation using LSTM models and internally used sequential model Sequence-to-Sequence (Seq2Seq), which is based on two LSTM sequential() networks: one for encoding the source language (input sentence) and the other for decoding into the target language (output sentence).

Input Embedding: The English input sequence is first converted into a sequence of dense vectors using an embedding layer. This layer learns to represent each word with a fixed-size vector.

LSTM Layers: One or more LSTM (Long Short-Term Memory) layers process the embedded input sequence. LSTMs are recurrent neural networks that are well-suited for handling sequential data. They capture the context and dependencies between words in the input sentence. it has some limitations.

Context Vector: The final hidden state and cell state of the last LSTM layer in the encoder are taken as the "context vector." This vector encapsulates the information from the entire input sentence.

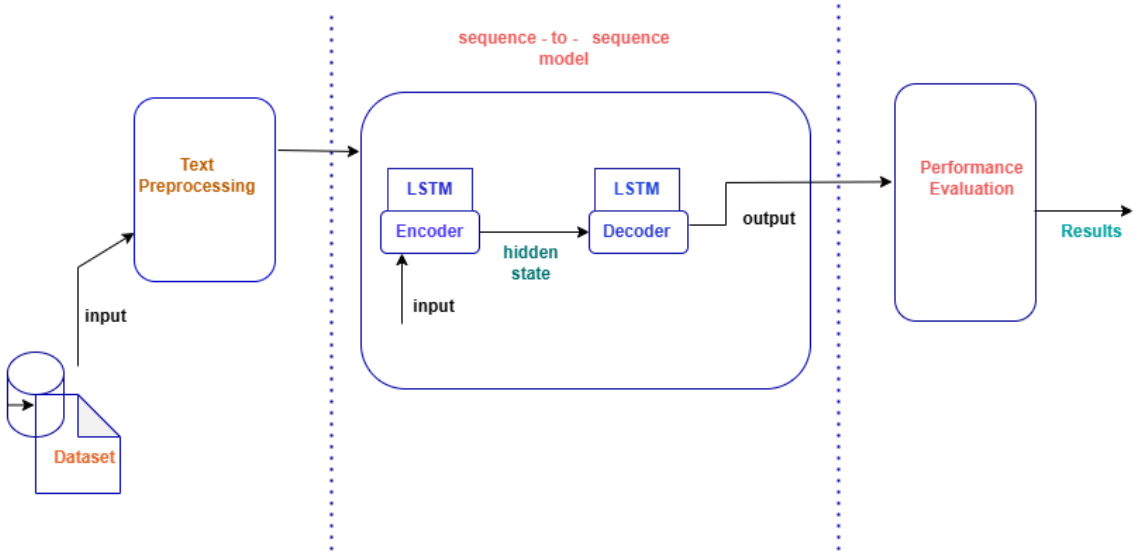

Fig. 2 – Language Translator Architecture

LSTM -Long Short-Term Memory, which is type of recurrent neural network a specialized type of RNN. LSTMs are designed to effectively capture both long-term and short-term dependencies within sequential data

LSTM effectively mitigates the problem of vanishing gradients and exploding gradients by introducing three gating mechanisms: the forget gate, the input gate, and the output gate, which enhances the model's ability to learn long term dependencies. The forget gate in the LSTM is responsible for deciding which information should be discarded from the cell state. The input gate determines which parts of new information should be stored, while the output gate controls which parts of the cell state will be outputted. Finally model stops the training data process when the errors are minimized. If we continuously train the model there is chances to occur exploding gradients.

LSTM Model.

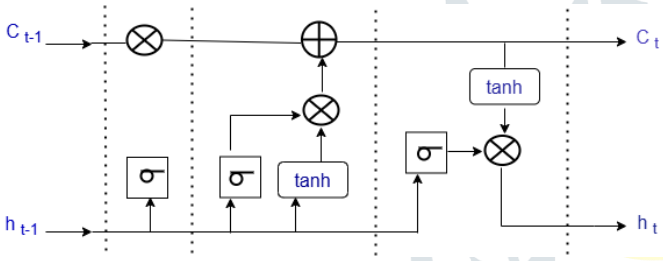

Fig. 3 – LSTM Model

Key Features of LSTMs and Components:

1. **Memory Cells**: These are responsible for retaining information over time, helping the model remember long-term dependencies.

Gates: LSTMs use gates (input, forget, and output) to control the flow of information. These gates decide what information should be remembered, forgotten, or passed to the next time step. LSTM gates regulate the flow of information into and out of the memory cel.

- **Forget Gate**: Determines what information from the previous cell state should be discarded. The forget gate decides what information from the previous cell state C_{t-1} will be discarded or kept. It uses a sigmoid activation function:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t]+b_f)$$

Where:

- W_f is the weight matrix for the forget gate.
- b_f is the bias for the forget gate.
- x_t is the input at time step ttt.
- h_{t-1} is the hidden state from the previous time step.
- σ is the sigmoid activation function.

- **Input Gate**: Decides what new information should be stored in the cell state. discover which value from input should be used to modify the memory. Sigmoid function decides which values to let through 0,1

The input gate controls which values will be updated in the cell state, and the candidate cell state is the new information to be added to the cell state.

Input Gate (i_t)

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t]+b_i)$$

- **Output Gate**: Controls the output of the current cell state to the next layer.

$$ot=\sigma(Wo \cdot [ht-1,xt]+bo)$$

where bo – bias, Wo- weight.

Sigmoid function

Sigmoid is Non Linear Activation Function

- Output range: 0 to 1 useful for binary classification
- Used in the "forget gate", "input gate", and "output gate" to decide how much information to pass through
- **Sigmoid:** Acts as a gating mechanism, controlling the flow of information by selectively allowing or blocking the input data.
- Provides a efficient transition between keeping and forgetting information

Tanh function

- Tanh is Non Linear Activation Function

- **Range:** Tanh outputs values range -1 and 1.
- Tanh is used to create a vector of new information values that could be added to the cell state.

These information values represent potential updates to the LSTM's memory.

Language translation using LSTM models typically involves an architecture called **Sequence-to-Sequence (Seq2Seq)**, which is based on two LSTM networks: one for encoding the source language (input sentence) and the other for decoding into the target language (output sentence).

System Design - LSTM Encoder Decoder Language Translator English to German

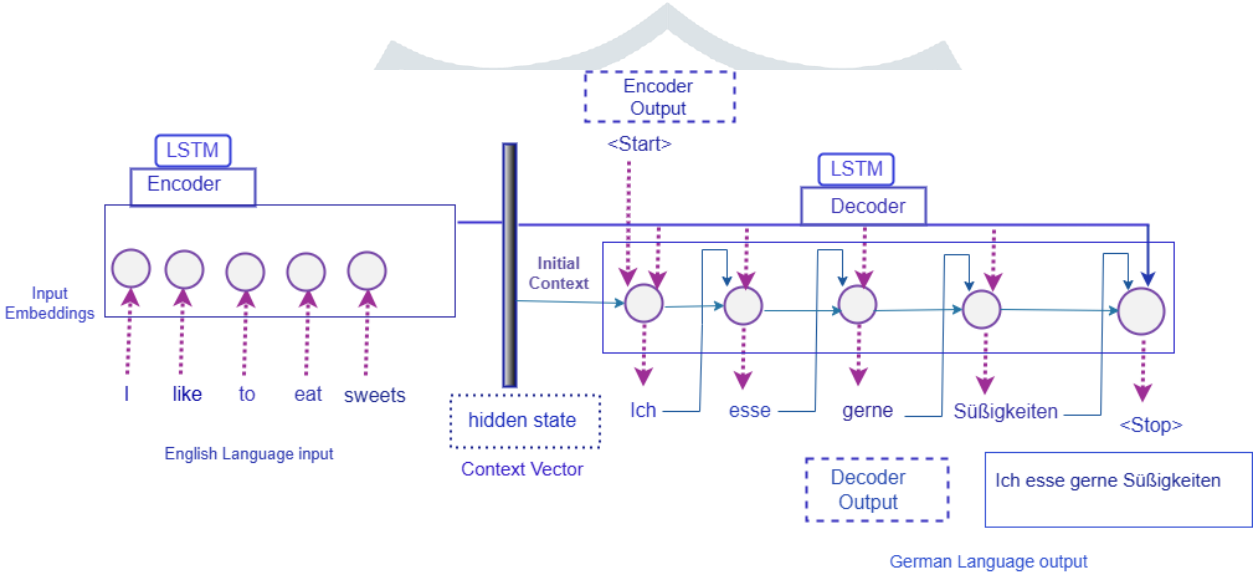

Fig. 4 - System Design - LSTM Encoder Decoder Language Translator English to German

For encoder-decoder models involving Seq2Seq (Sequence to Sequence) Architecture. it is basically used for response generation whereas in machine translation models it is used to find the relationship between two different language pairs. It consists of two parts, an encoder and a decoder. The encoder takes the input from source and the decoder generates the output based on input from encoding vector previously generated words.

Components of a Seq2Seq LSTM for Language Translation:

Encoder:

- The encoder is usually an LSTM network that processes the input sentence word-by-word (or token-by-token). As it reads the input sequence, it encodes the sentence into a fixed-size context vector (also called the **hidden state**) Mapping the input sequence with target language and put in context vector in encoded format.
- This context vector is supposed to capture the essential information from the input sentence, such as its grammatical structure and meaning.

Decoder:

- The decoder is another LSTM network that generates the translated sentence in the target language. It receives the context vector from the encoder and starts generating words for the output sequence.

- At each step, the decoder generates a word (or token) and uses that word as input for generating the next word. During training, this is typically done using **teacher forcing**, where the true real outout previous word is fed into the decoder.

IV. Experiment

We start by demonstrating the Europal datasets is used, It takes the sentence and do preprocessing steps and process word by word and brings into the context vector by LSTM Encoder in the hidden state, back propagation is done and adjust updating the parameters weights and fine tune the model and LSTM Decoder takes the input passed by encoder and generate the output. and finally, we show the experimental results

For training the model, we have used the Python 3 Google Compute Engine backend (TPU) provided by Google Colab, We used the LSTM embedding then We compile the model and configured the batch size .

 # Compiling the model

```
model_translate_lstm.compile(loss=sparse_categorical_crossentropy, optimizer=Adam(1e-3), metrics=['accuracy'])
```

 # fit the model

```
history = model_translate_lstm.fit(X_train, Y_train, epochs=10, batch_size=128, validation_split=0.3, callbacks=[es, rl, mc])
```

to be equal to 128 records and the effective batch size which is defined as the number of training examples consumed in one step. We optimized the model parameters using the Adam optimizer. and a learning rate of 0.0005. The models were trained on 10 epochs each for every language pair.

Epoch 9: val_accuracy did not improve from 0.89096

31s 813ms/step - accuracy: 0.8871 - loss: 0.9417 - val_accuracy: 0.8864 - val_loss: 0.9443 - learning_rate: 0.0010

Epoch 10/10

0s 534ms/step - accuracy: 0.8859 - loss: 0.9381

Epoch 10: val_accuracy did not improve from 0.89096

41s 804ms/step - accuracy: 0.8859 - loss: 0.9380 - val_accuracy: 0.8909 - val_loss: 0.9322 - learning_rate: 0.0010

we were able to get the BLEU score, which was obtained by comparison between the translated sentences by the model from the source language, with the corresponding target language translations.

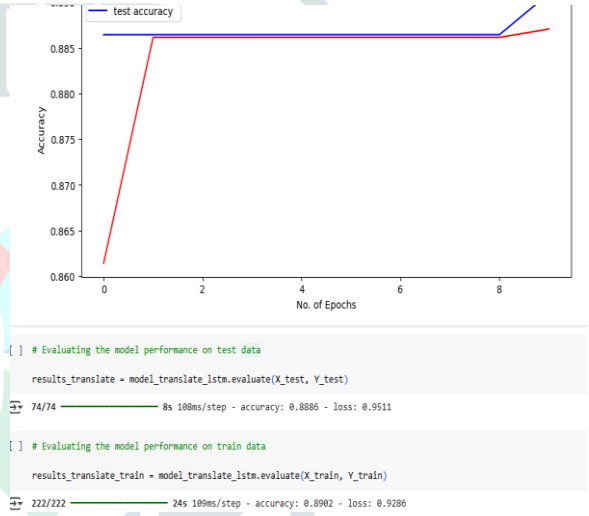

Fig . 5 - Model Evaluation Performance BLEU

V. Challenges in existing application

English to Germany translation was attempted using LSTM neural network and a highly limited vocabulary corpus dataset. the limited parallel corpus by back- and forward translation. Quality of Service results depends on the size and quality of the corpus, and translations can sometimes be unnatural or incorrect. Requires large datasets and computational power, and there may still be errors with complex grammar or specialized terms.

Domain-Specific Language: German to English Translating highly specialized fields like Europian Parliment conversations.

VI. Multi Language Translation System

Translating text from source language to target language we integrate Google Neural Machine Translation. Google NMT system uses a large artificial neural network capable of deep learning. By using millions of examples, Google NMT improves the quality of translation increase fluency and accuracy

One to Many Translation

Google's Multilingual Neural Machine Translation (NMT) model is a single model that can translate between multiple languages. It uses a shared vocabulary and an artificial token at the start of a sentence to specify the target language

- The model shares parameters across all language pairs, which helps it generalize across language boundaries
- It can improve translation quality for all language pairs involved and It can improve translation quality for low-resource language pairs
- It can perform zero-shot translation
- It can perform implicit bridging between language pairs

Most of the commercial machine translation systems in use to developed using a rules-based approach and require a lot of work by linguists to define vocabularies and grammars. Several research systems, Google NMT takes a different approach: feed the computer with billions of words of text, both monolingual text in the target language, and aligned text consisting of examples of human translations between the languages. then apply statistical learning techniques to build a translation model

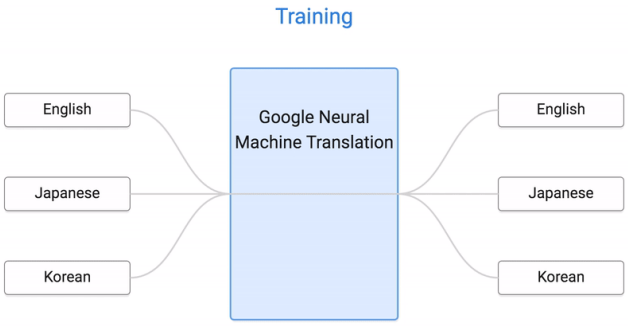

Fig .6 – Multi Language Translator NMT

Working Principle

Uses Attention Mechanism captures the dependencies gives the importance to the words and brings into context vector, and uses multihead attention transformer for parallel computation.

$$MultiHead(Q,K,V)=concat(head1.head2\ldots headn)WO$$

The encoder start by processing the input sequence. The output of the top encoder is then transformed into a set of attention vectors K and V. These are to be used by each decoder in its "encoder-decoder attention" layer which helps the decoder focus on appropriate places in the input sequence:

Scaled-dot product Matrix Calculation of Self-Attention

Calculate the Query, Key, and Value matrices. takes our embeddings into a matrix X, and multiplying it by the weight matrices we've trained (WQ, WK, WV).

in order to calculate multi-head attentions, we prepare 8 pairs of "queries", "keys" , and "values", We calculate attentions and reweight "values" independently in 8 different heads, and in each head the reweighted "values" are calculated with this very simple formula of scaled dot-product:

$$Attention(Q,K,V)=softmax(\ (QK^{T})\ /\sqrt{d_k}\ )V.$$

Sample Translation results:

Input Language (English): Good Morning

Translated Text (Tamil): காலை வணக்கம்
Translated Text (hindi): शुभ प्रभात
Translated Text (German): Guten Morgen
Translated Text (French): Bonjour
Translated Text (Spanish): Buen día

Translated Text (Malayalam): സുപ്രഭാതം
Translated Text (Chinese): 早上好

input_text_Tamil ::::: காலை வணக்கம்
Translated Text Tamil To English :::::: (Tamil To English): Good morning

Evaluation BLEU score.

BLEU Score (Tamil): 0.7175852914772134
BLEU Score (Hindi): 0.6389431042462724
BLEU Score (German): 0.6786502681586727
BLEU Score (French): 1.0
BLEU Score (Spanish): 0.5
BLEU Score (Malayalam): 1.0
BLEU Score (Chinese): 1.2213386

Average BLEU Score (Model Performance): 0.64

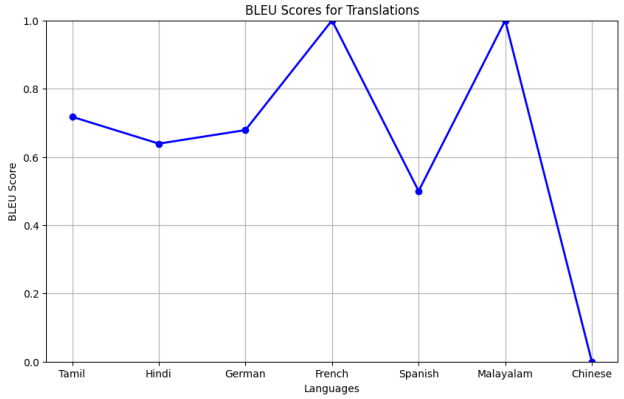

Fig.7 – Multi Language Translator Model Performance Evaluation BLEU score

VII. Conclusion

In this paper represented a single language model translation system to multilingual NMT. We show that we can train multilingual NMT model and it can be used to translate between a number of different languages using a single model where all parameters are shared, and model learning process shows that zero-shot translation which is that a form of true transfer learning has been shown to work for machine translation. To explicitly improve the zero-shot translation quality. and model

performance we achieved a BLEU Score Average score value of 0.64 .

Implementation and the results shows that these models learn a form of interlingua representation between all involved language pairs. The simple architecture makes it possible to mix languages on the source or target side to produce some interesting translation output. Our approach has been shown to work reliably in a Google-scale production setting and enables us to scale to a large number of languages quickly.

VIII. References

- Martin Abadiand Paul Barhametal. 2016. Tensorflow: A system for large-scale machine learning. In 12thUSENIX Symposiumon Operating Systems Design and Implementation (OSDI16), pages265–283.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015.Neural machine translation by jointly learning To align and translate. In International Conference on Learning Representations.
- Alfred V. Aho and Jeffrey D. Ullman. 1972. The Theory of Parsing, Translation and Compiling, volume 1. Prentice-Hall, Englewood Cliffs, NJ.
- F. Stahlberg, ―Neural Machine Translation: A Review,‖ J. Artif. Intell. Res., vol. 69, pp. 343–418, 2020.
- H. Wang, H. Wu, Z. He, L. Huang, and K. W. Church, ―Progress in machine translation,‖ Engineering (Beijing), 2021.