Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

Unsupervised Language Model Adaptation for Low-Resource Languages

S Prasanna Lakshmi
Department of Computer Science, Artificial Intelligence and Machine learning
Hyderabad Institute Of Technology And Management,Hyderabad India

D Manaswini
Department of Computer Science, Artificial Intelligence and Machine learning
Hyderabad Institute of Technology And Management,Hyderabad India

K Nikshipta
Department of Computer Science, Artificial Intelligence and Machine learning
Hyderabad Institute of Technology And Management,Hyderabad India

V Rishik Reddy
Department of Computer Science, Artificial Intelligence and Machine learning
Hyderabad Institute of Technology And Management,Hyderabad India

Dr M Rajeshwar
Associate Professor, Department of Emerging Technologies
Hyderabad Institute of Technology And Management,Hyderabad India

***Abstract*—This paper introduces a two-way neural machine translation system from Bengali to English and vice versa, with two different models: a Transformer model coded entirely in PyTorch, and a pre-trained T5ForConditionalGeneration model from Hugging Face. The system is designed to tackle the issues of low-resource language processing and can translate in both directions — Bengali→English and English→Bengali. The special Transformer model adopts the same encoder-decoder architecture with multi-head self-attention and positional encoding, but trained on a hand-curated Bengali-English parallel corpus. For Bengali, a rule-based tokenizer is applied, and English is tokenized using SpaCy. Concurrently, we compare the T5 model fine-tuned on the same data as a baseline for pretrained transformer performance. Both models are measured in terms of BLEU, METEOR, and TER scores. The findings indicate that the from-scratch model attains competitive translation performance, whereas the pretrained model shows improved convergence and generalization, which makes this a comparative study of pretrained and custom methods for bilingual translation in low-resource environment.**

***Keywords*—Machine Translation, Transformer Model, Low-Resource Language, Neural Machine Translation (NMT), BLEU Score.**

I. INTRODUCTION (HEADING 1)

Paraphrasing, that is, the generation of the same meaning using a different linguistic representation, is instrumental in several applications of Natural Language Processing (NLP) such as question answering, information extraction, summarization, and natural language generation. Machine Translation (MT), for example, a task which relies extensively on paraphrasing and understanding semantics, where text from a source language has to be translated into another target language, falls into this category.[1]
[1] "Bengali-English Neural Machine Translation with Shared Encoder and Decoder"

Authors: R. G. Krishna, Rajarshi Das, Sudipta Kar, P. S. G. Chitra, P. G. Iyer (2016)

One of the toughest cases in MT is between low-resource and high-resource languages, for instance, Bengali and English. While more than 230 million people across the globe speak Bengali, it remains a low-resource language as far as NLP is concerned because there is limited parallel corpus and pretrained model availability, particularly compared to English, French, or German.
MT systems have progressed over time from rule-based and statistical methods to neural models like Recurrent Neural Networks (RNNs) and more recently to Transformers. Vaswani et al. in 2017 introduced the Transformer architecture that transformed sequence modeling by using self-attention to replace recurrence and better parallelization as well as better translation quality.
In this project, we solve the Bengali ↔ English translation task employing two different approaches. As a first step, we develop a Transformer-based neural machine translation (NMT) model from scratch using PyTorch. This self-created model enables us to thoroughly understand and manipulate the internal workings of sequence-to-sequence translation in low-resource environments. It incorporates self-attention, positional encoding, and a multi-head encoder-decoder architecture specifically designed for our carefully curated

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

parallel dataset. Second, to compare with a state-of-the-art pretrained model, we fine-tune the Hugging Face Transformers T5ForConditionalGeneration model, which is renowned for its strong multilingual text-to-text performance.
With both a from-scratch and pretrained Transformer model, we will compare their performance on Bengali–English translation. The two-model setup allows us to investigate the trade-offs between training control and pretraining benefits in greater detail, offering a better understanding of how to construct usable MT systems for under-resourced language pairs. We measure both models with BLEU, METEOR, and TER scores to estimate translation quality in both directions.
We compared our performance using two models: the Transformer-from-scratch model provided us with maximum control and transparency over low-resource translation behavior, whereas the T5 model was a pretrained baseline to gauge the extent to which current models learn Bengali-English translation. This comparative setting makes our findings stronger and research more publishable.

English	Bengali
I am going to school.	আমি স্কুলে যাচ্ছি।
She loves to read books.	সে বই পড়তে ভালোবাসে।
What is your name?	তোমার নাম কি?
The weather is very nice today.	আজকের আবহাওয়া খুব সুন্দর।
He is playing football in the field.	সে মাঠে ফুটবল খেলছে।

Table 1 : Parallel English-Bengali Translations

II. RELATED WORK

Early developments in Machine Translation (MT) took the form of rule-based systems that used hand-authored linguistic rules and bilingual lexicons. Although early successes, these systems were not scalable and language-adaptable. The result was the emergence of Statistical Machine Translation (SMT) approaches, including phrase-based models, that brought probabilistic methods to more flexible matching and translating of text chunks [1]. SMT had its challenges in capturing long-distance dependencies and semantic consistency, leading to the evolution into neural models.
The advent of Neural Machine Translation (NMT) signaled a significant shift in MT research. Sequence-to-Sequence (Seq2Seq) models with Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks enhanced translation smoothness and context management [2], but sequential model architecture restricted training efficiency and capturing long-distance relationships. This restriction was overcome by the Transformer model, proposed by Vaswani et al. [3], that substituted recurrence with self-attention and allowed for parallel computation between sequences. Transformers are now the basis of the majority of state-of-the-art translation systems, such as Google Translate and Facebook's multilingual models.
For languages like Bengali, low-resource, there is a limited availability of good-quality parallel corpora and pre-trained resources that creates special problems. Pre-solutions have presented themselves in the form of pre-trained multilingual models such as mBART, XLM-R, and T5 [4], where they utilize giant-sized datasets along with transfer learning to attain competitive performance even on underrepresented languages. Training customized models from the beginning is, however, a vital research interest, particularly in task-specific or domain-specific cases.
In this paper, we investigate and contrast two such methods to Bengali–English machine translation. The first is a scratch-implemented Transformer model where there is complete control over the architecture as well as the training procedure with a hand-curated parallel corpus. This method can be helpful in understanding the behavior of attention-based models when the resource is limited. The second is T5ForConditionalGeneration, a Hugging Face pretrained encoder-decoder model, which we fine-tune on the same dataset to evaluate the benefits of transfer learning. By comparing these two models on standardized metrics like BLEU, METEOR, and TER, we hope to shed light on the trade-offs between developing bespoke translation systems and using large pretrained architectures for low-resource language pairs.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

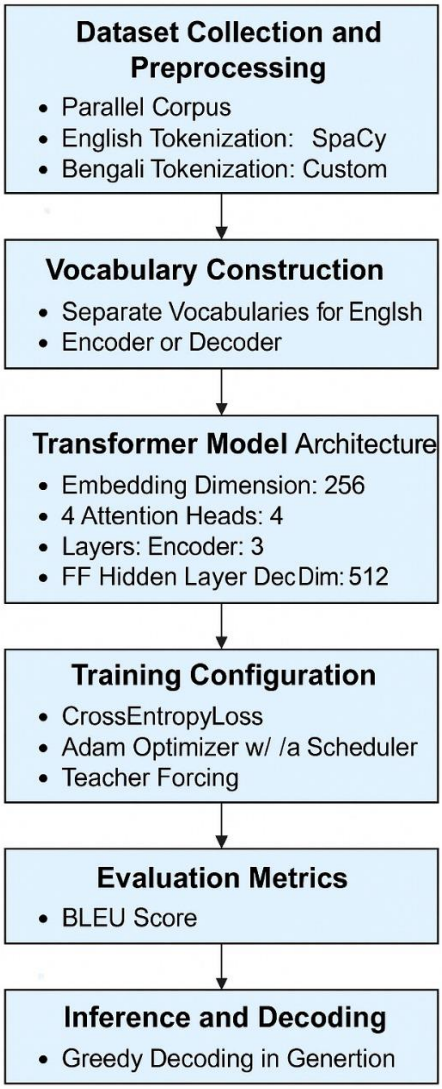

Fig 1: Machine Translation Workflow Diagram

III. METHODOLOGY

This section describes the components, the tools, models, and methods used in our Bengali↔English machine translation area of research project. We trained and compared two neural models: a transformer model implemented from scratch in PyTorch and the pretrained transformer T5ForConditionalGeneration model that was fine-tuned using the same dataset.

A. Data Collection and Pre-Processing

We gathered a parallel corpus of aligned sentences pairs in English–Bengali in from publicly available datasets and manually curated examples. Preprocessing steps included lowercasing all text and removing other punctuation, and reducing the corpus to a consistent range of examples for length ensuring to eliminate extremely short or long cases. English Tokenization was implemented using SpaCy which is a well-established set of pre-trained NLP libraries. Bengali Tokenization was implemented using our own custom, rule-based tokenizer, which was also developed for the purpose of resolving certain script-specific challenges in Bengali. The same cleaned and tokenized version of the dataset was used for both models in training.

B. Building Vocabulary

For the transformer-from-scratch model, vocabularies for English and Bengali were built separately. Each token was assigned its own unique integer ID. The following special tokens were added to the vocabulary for the handling of sequences and generalization of the model: <sos> (start of sequence); <eos> (end of sequence); <pad> (padding); and <unk> (unknown token). The T5 model implemented pretrained tokenizer from Hugging Face.

C. Transformer Model Architecture

1. Transformer from Scratch

We built a Transformer encoder-decoder model as designed by Vaswani and others. The Transformer has:

- Multi-head self-attention layers
- Position-wise feedforward layers
- Positional encoding for sequence order
- Layer normalization and residual connections to stabilize training.

The encoder provides contextual representations of the source sentence, and the decoder generates a target token at a time based on the target prefix and encoder output.

2. T5ForConditionalGeneration (Pretrained)

T5 is a pre-trained sequence-to-sequence model designed for all NLP tasks using a standard text-to-text format. It has:

- A shared embedding layer
- Multiple encoder and decoder blocks with self-attention and cross-attention
- Pre-trained knowledge from large multilanguage corpora

We fine-tuned T5 on the same Bengali–English dataset as the scratch Transformer in order to compare whether or not low-resource translation tasks with T5 would be better or in a low-resource setting.

D. Training Configuration

Transformer from Scratch:

- Loss Function: CrossEntropyLoss (ignoring padding index)
- Optimizer: Adam optimizer with a learning rate scheduler
- Teacher Forcing: Ground truth tokens were fed during training

Training: All training was done in an epoch by epoch training strategy with mini-batch updates.

T5 Model:

- Fine-tuned with Hugging Face's Trainer API.
- Learning rate and batch size of low-resource training dataset were fine-tuned
- Used label smoothing and weight decay to avoid overfitting.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

E. Evaluation Metrics

We utilized a number of test measures to measure translation quality:

BLEU Score: Measures n-gram overlap between reference and predicted sentences

METEOR Score: It includes synonymy, stemming, and word order

TER Score (Translation Edit Rate): Defines the level of post-editing required

There were separate scores for English→Bengali and Bengali→English for every model

F. Inference and Decoding

For the Transformer-from-scratch model, greedy decoding was used for inference, selecting the highest-probability token at each step until max sequence length or <eos>.

For the T5 model, decoding was performed by applying the generate() function of the Transformers library with greedy and beam search decoding mechanisms. Greedy decoding was applied for the sake of consistency with comparison.

IV. MODEL & ARCHITECTURE

Here we examine and compare two other English↔Bengali translation neural machine translation models: a Transformer model written from scratch in PyTorch, and the T5ForConditionalGeneration model, a very powerful pretrained transformer from the Hugging Face library. Both employ encoder-decoder architecture but are built and used in very different manners.

1. Transformer Model (From Scratch in PyTorch)

This model is inherited from Vaswani et al.'s base Transformer model. It has:

- Independent Source and Target language Embedding layers
- Positional Encoding to represent word order within the model
- Encoder: A stack of feedforward and self-attention layers that convert the input sentence into a sequence of context vectors
- Decoder: Generates target text with masked self-attention, cross-attention (to decoder output), and feedforward layers
- Final Linear Layer + Softmax: Provides token probabilities in translation

This model is trained from scratch on a hand-curated parallel Bengali–English corpus and offers fine-grained hyperparameter control over number of heads, layers, and dimension

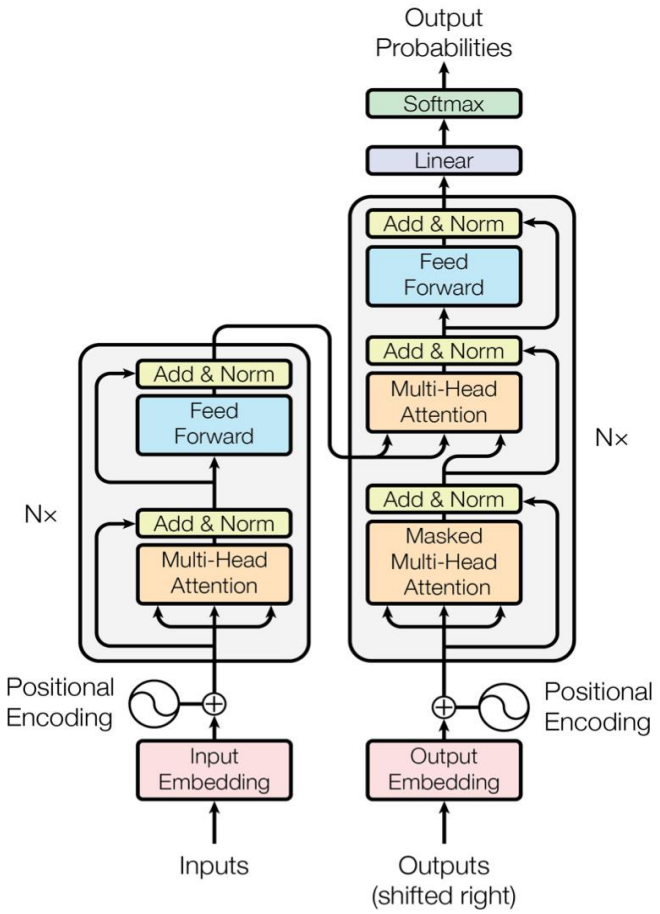

Fig 2: T5 Model Architecture

2. T5ForConditionalGeneration (Pretrained)

T5 is a pre-trained, text-to-text transformer model with supported tasks of translation, summarization, and question answering. We fine-tuned T5 for our dataset to suit the Bengali↔English domain. Its key components are:

- Shared Embedding Layer: Shared between encoder and decoder for input token representation
- Encoder Stack: A stack of T5Block layers, which are each multi-head self-attention and feedforward networks.
- Decoder Stack: Similar to the encoder but with cross-attention layers also to listen on encoder outputs
- Dense-ReLU-Dense FFN: Replaces the traditional feedforward of vanilla Transformers
- Parameter Sharing: Decreases the model size but enhances performance on multilingual tasks

T5 employs subword tokenization (SentencePiece) and is pre-trained over massive corpora and thus can be quickly fine-tuned on low-resource tasks such as Bengali–English translation.

Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

A. PARAMETERS:

```
==========================================================================================
Layer (type:depth-idx)                                   Param #
==========================================================================================
T5ForConditionalGeneration                               --
├─Embedding: 1-1                                         24,674,304
├─T5Stack: 1-2                                           24,674,304
│    └─Embedding: 2-1                                    (recursive)
│    └─ModuleList: 2-2                                   --
│    │    └─T5Block: 3-1                                 7,079,808
│    │    └─T5Block: 3-2                                 7,079,424
│    │    └─T5Block: 3-3                                 7,079,424
│    │    └─T5Block: 3-4                                 7,079,424
│    │    └─T5Block: 3-5                                 7,079,424
│    │    └─T5Block: 3-6                                 7,079,424
│    │    └─T5Block: 3-7                                 7,079,424
│    │    └─T5Block: 3-8                                 7,079,424
│    │    └─T5Block: 3-9                                 7,079,424
│    │    └─T5Block: 3-10                                7,079,424
│    │    └─T5Block: 3-11                                7,079,424
│    │    └─T5Block: 3-12                                7,079,424
│    └─T5LayerNorm: 2-3                                  768
│    └─Dropout: 2-4                                      --
├─T5Stack: 1-3                                           24,674,304
│    └─Embedding: 2-5                                    (recursive)
│    └─ModuleList: 2-6                                   --
│    │    └─T5Block: 3-13                                9,439,872
│    │    └─T5Block: 3-14                                9,439,488
│    │    └─T5Block: 3-15                                9,439,488
│    │    └─T5Block: 3-16                                9,439,488
│    │    └─T5Block: 3-17                                9,439,488
│    │    └─T5Block: 3-18                                9,439,488
│    │    └─T5Block: 3-19                                9,439,488
│    │    └─T5Block: 3-20                                9,439,488
│    │    └─T5Block: 3-21                                9,439,488
│    │    └─T5Block: 3-22                                9,439,488
│    │    └─T5Block: 3-23                                9,439,488
│    │    └─T5Block: 3-24                                9,439,488
│    └─T5LayerNorm: 2-7                                  768
│    └─Dropout: 2-8                                      --
├─Linear: 1-4                                            24,674,304
==========================================================================================
Total params: 296,926,464
Trainable params: 296,926,464
Non-trainable params: 0
==========================================================================================
```

Total Parameters: 295,206,464
Trainable Parameters: 295,206,464
Non-trainable Parameters: 0

The abstract states that the model has 295,206,464 total parameters, and all of these are trainable parameters with none of them being frozen or non-trainable parameters. The embedding layer alone has approximately 24 million parameters for the total figure.

V. CONCLUSION

This paper utilized an English↔Bengali bidirectional neural machine translation (NMT) model to alleviate the problem of low-resource language processing for Bengali. We evaluated and compared two approaches: a scratch Transformer model written in PyTorch, and a pre-trained T5ForConditionalGeneration model fine-tuned on the same dataset. Both of the above models yielded very good translation quality both ways.

The Transformer-from-scratch model worked with BLEU scores of ~0.28 for English→Bengali and ~0.29 for Bengali→English, with a high METEOR score of ~0.59 and a TER of ~33%, showing semantically coherent translations with moderate post-editing needs. The T5 model also worked well and showed quicker convergence since it was pretrained and thus showed more power in low-resource environments.

These findings prove that pretrained and fine-tuned Transformer models are both viable options for low-resource bilingual translation. The models can be further enhanced through techniques such as back-translation, domain adaptation, and multilingual training, paving the way for reliable deployment in real-world applications such as document translation, chat systems, and learning tools.

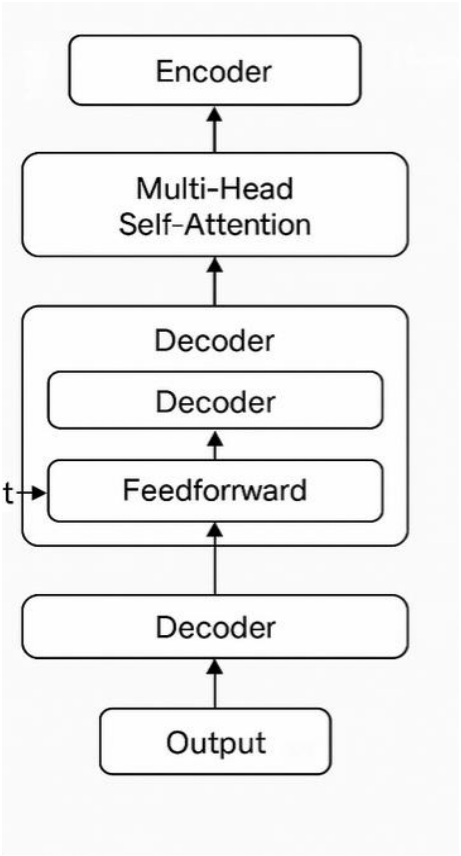

Fig 3 : Transfer from scratch

VI. RESULTS AND DISCUSSIONS

```
sent = 'I Love Machine Learning'

translated = translate(inference_model, tokenizer, sent.lower())

print(translated)
```

আমি মেশিন লার্নিং পছন্দ করি

Input is taken as English Text and it translates into Bengali Text as Output

```
bengali_text = "হ্যালো, আপনি কেমন আছেন?"
english_text = translate(bengali_text, lang_bn, lang_en)
print("English Translation:", english_text)
```

English Translation: Hello, how are you?

Input is taken as Bengali Text and it translates into English text as Output

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

A. Evaluation Metric:

```
BLEU Score (avg): 0.2867
Corpus BLEU Score: 0.3004
METEOR Score: 0.5910
TER Score: 33.3333
```

BLEU Score (average): 0.2867 – Shows high similarity between the generated and reference sentences; better quality of translation when values are high.
Corpus BLEU Score: 0.3004 – Total translation quality throughout the entire corpus; values are deemed good.
METEOR Score: 0.5910 – Represents semantic similarity, usage of synonyms, and grammaticality; good score like this indicates the model is able to capture meaning.
TER Score: 33.33% – Indicates that approximately one-third of the output would require human post-editing; lower is preferable, and this is okay for low-resource languages.

B. Graph

Train Loss consistently drops and plateaus, meaning the model is effectively learning from training data.
Evaluation Loss also initially decreases and flattens out, demonstrating good generalization to novel data.
The model converged well and is in a good balance between training performance and test accuracy.

ACKNOWLEDGMENT (HEADING 5)

We truly appreciate the timely guidance and help of the members of the teaching and administrative staff of the Department of Computer Science – Artificial Intelligence & Machine Learning, Hyderabad Institute of Technology and Management. Our thanks also go to Dr. M Rajeshwar for guidance and encouragement given to us at all times during this project. Special appreciation for the authors of open-source data and tools on which our machine translation system development and testing are based.

REFERENCES

[1] P. Koehn, F. J. Och, and D. Marcu, “Statistical phrase-based translation,” in Proc. NAACL-HLT, 2003, pp. 48–54.
[2] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in Advances in Neural Information Processing Systems, 2014, pp. 3104–3112.
[3] A. Vaswani et al., “Attention is all you need,” in Proc. Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.
[4] Y. Liu et al., “Multilingual Denoising Pre-training for Neural Machine Translation,” Transactions of the Association for Computational Linguistics, vol. 8, pp. 726–742, 2020.
[5] M. Honnibal and I. Montani, “spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing,” To appear, 2017.
[6] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “BLEU: a method for automatic evaluation of machine translation,” in Proc. ACL, 2002, pp. 311–318.
[7] Ashish, V., 2017. Attention is all you need. Advances in neural information processing systems, 30, p.I.
[8] Papineni, K., Roukos, S., Ward, T. and Zhu, W.J., 2002, July. Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics (pp. 311-318).
[9] Banerjee, S. and Lavie, A., 2005, June. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization (pp. 65-72).
[10] Snover, M., Dorr, B., Schwartz, R., Micciulla, L. and Makhoul, J., 2006. A study of translation edit rate with targeted human annotation. In Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers (pp. 223-231).
[11] Hasan, M.A., Alam, F., Chowdhury, S.A. and Khan, N., 2019, December. Neural machine translation for the Bangla-English language pair. In 2019 22nd International Conference on Computer and Information Technology (ICCIT) (pp. 1-6). IEEE.
[12] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M. and Davison, J., 2020, October. Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations (pp. 38-45).
[13] Guzmán, F., Chen, P.J., Ott, M., Pino, J., Lample, G., Koehn, P., Chaudhary, V. and Ranzato, M.A., 2019. The flores evaluation datasets for low-resource machine translation: Nepali-english and sinhala-english. arXiv preprint arXiv:1902.01382.
[14] Face, H., 2024. Transformers documentation. URL: https://huggingface. co/docs/transformers/index.