PROJECT REPORT

MACHINE TRANSLATION

ADIL SHAIK	SNEHA VANGALA	KOMAL TANKASHALA
MS IN DATA SCIENCE	*MS IN DATA SCIENCE*	*MS IN DATA SCIENCE*
Ashai45@unh.newhaven.edu	*Svang16@unh.newhaven.edu*	*ktank1@unh.newhaven.edu*

***ABSTRACT*:** This paper introduces a new approach to machine translation from English to Hindi, Employing technequies in Natural Language processing and Deep Learning . The proposed model addresses the challenges inherent in cross-language translation by integrating advanced methodologies. The paper provides a comprehensive overview, beginning with the motivation for the research and progressing through technical details, training optimization, and thorough evaluation. The model's performance is evaluated using standard metrics, in-cluding the BLEU score, on a dedicated test set. Comparative analyses with baseline models showcase the effectiveness of the proposed methodology, while discussions on successes and challenges contribute valuable insights to the field of machine translation.

KEYWORDS: LSTM, CNN, TCN, F1 score, accuracy

I. INTRODUCTION:

The primary objective of this project is to push the boundaries of machine translation, particularly for the English-to-Hindi language pair, with the ultimate goal of enhancing global communication by breaking down language barriers. The motivation stems from the increasing demand for accurate and contextually meaningful translations between these languages, which is crucial for fostering effective cross-cultural communication and facilitating seamless information sharing on a global scale. Leveraging recent breakthroughs in natural language processing (NLP) and deep learning, the project integrates cutting-edge methodologies to improve translation accuracy and fluency. By harnessing the power of advanced techniques and models, the project aims to overcome the limitations of existing systems and establish new benchmarks in machine translation performance. This endeavor holds significant promise for facilitating smoother communication and interaction across diverse linguistic and cultural backgrounds, ultimately contributing to a more connected and inclusive global community.

The primary goals of the project include:

- Capturing the nuances of both English and Hindi languages, ensuring accurate and culturally appropriate translations.

- Optimization through Hyper parameter Tuning: Employing systematic optimization strategies, including hyper parameter tuning, to enhance the training process and overall performance of the model.

- Evaluation Using Standard Metrics: Assessing the quality of translations using standard metrics such as the BLEU score, providing a quantitative measure of the model's proficiency.

- Comparative Analysis: Comparing the proposed model against baseline models to highlight the advancements achieved and benchmark the system's effectiveness.

- Throughout this journey, the project seeks to contribute not only to the specific domain of English-to-Hindi translation but also to the broader landscape of machine translation research. By addressing challenges, exploring innovative methodologies, and presenting comprehensive analyses, the project endeavors to make meaningful strides in advancing the capabilities of machine translation systems.

II. RELATED WORK:

This paper explores the application of sequence-to-sequence (Seq2Seq) models in the realm of machine translation, with a particular emphasis on translating English to Hindi. Seq2Seq models, initially introduced by Sutskever et al. (2014), form a cornerstone of contemporary machine translation systems. The subsequent review delves into the current body of research dedicated to investigating and refining the Seq2Seq architecture for English-to-Hindi translation.

1. Baseline Sequence-to-Sequence Models: The seminal work by Sutskever et al. laid the foundation for Seq2Seq models, employing recurrent neural networks (RNNs) for sequence encoding and decoding. Despite its effectiveness, this early approach faced challenges in capturing long-range dependencies and context.

2. Attention Mechanisms for Improved Alignment: The integration of attention mechanisms into Seq2Seq models was proposed by Bahdanau et al. (2015), enabling the model to selectively focus on various segments of the source sequence while decoding. This introduction of attention mechanisms resulted in a notable improvement in the alignment between source and target language words, consequently enhancing translation accuracy.

3. Transformer Models and Self-Attention: A pivotal moment in Seq2Seq models occurred with the introduction of the Transformer architecture by Vaswani et al. (2017). Transformers leverage self-attention mechanisms, enabling the model to weigh the importance of different words in the input sequence simultaneously. This innovation efficiently addressed long-range dependencies, leading to a substantial improvement in translation quality.

4. Transfer Learning with Pre-trained Models: The advent of transfer learning strategies, exemplified by models like BERT (Devlin et al., 2018), has gained prominence. Pre-training models on extensive datasets facilitates a better contextual understanding. Researchers have explored the applicability of pre-trained models for English-to-Hindi translation, demonstrating promising results in capturing nuanced contextual information.

5. Domain Adaptation and Specialized Translation:To tackle the challenges of domain-specific translations, studies have delved into domain adaptation techniques. Specialized translation tasks often require fine-tuning or adapting Seq2Seq models to specific domains, as exemplified by Bapna et al. (2019).

6. Data Augmentation and Back-Translation: Addressing data scarcity issues, researchers have explored data augmentation techniques such as back-translation. Back-translation involves translating target language sentences back to the source language and using them as additional training data, effectively augmenting the dataset and improving model robustness (Sennrich et al., 2016).

7. Evaluation Metrics and Challenges: While traditional metrics like BLEU (Papineni et al., 2002) are commonly used for evaluation, recent studies emphasize the need for more nuanced metrics aligned with human assessments. Challenges persist in handling idiomatic expressions, cultural nuances, and low- resource language pairs.

III. METHODOLOGY:

1. Data Collection: The data collection process involved manual scraping of information from the official Yale website, resulting in a dataset comprising 128 samples. The dataset was prepared for a machine translation task, with English as the input language and Hindi as the output language. To obtain translations, the Google Translate service

	English	
0	Yale offers advanced degrees through its Gradu...	येल अपने ग्रेजुएट स्कूल ऑफ आर्ट्स एंड साइ
1	Browse the organizations below for information...	अध्ययन के कार्यक्रमों, शैक्षणिक आवश्यकताओं
2	Graduate School of Arts & Sciences.	ग्रेजुएट स्कूल ऑफ आर्ट्स एंड स
3	Yale's Graduate School of Arts & Sciences offe...	येल के ग्रेजुएट स्कूल ऑफ आर्ट्स एंड साइंसेज
4	School of Architecture.	स्कूल ऑफ आर्कि
5	The Yale School of Architecture's mandate is f...	येल स्कूल ऑफ आर्किटेक्चर का जनादेश प्रत्येक

2. Data Processsing: The preprocessing phase aimed at creatingdictionaries for word-to-index, word-to-count, index-to-word, and vocabulary. Textual data underwent necessary transformations, including the removal of undesired special characters, conversion to lowercase, and elimination of trailing white spaces. Additionally, statistics on the total number of words in each language'svocabulary were computed.

3. Custom dataset Creation: To facilitate model training, a custom datasetclass was implemented. This class included a`_len__` method to determine the dataset lengthand a `_getitem_` method providing input and output sentence tensors along with their corresponding text representations based on an index. An illustrative example was provided, showcasing English and Hindi sentences alongside their respective tensor representations.

4. Data Loading: The dataset was split into training, testing, and validation subsets using the `train_test_split` method from sklearn. For efficient loading during model training, PyTorch's DataLoader was employed with a batch size of 8. To address imbalanced tensor sizes across samples, a collate function was implemented, incorporating padding techniques.

5. Data Overview: The dataset comprised 128 samples, with English as the input language and Hindi as the output language. Vocabulary sizes were calculated for both languages. During training, batches of 8
samples were loaded using PyTorch DataLoader, with a collate function mitigating imbalanced tensor sizes through padding techniques.

6. For this project, a pretrained model was utilized. Named Marian MTModel, this model comprises an encoder-decoder architecture with shared embeddings. The embedding layer possesses a vocabulary size of 61,950, where each token is depicted by a 512-dimensional vector. The encoder is equipped with six Marian Encoder Layers, each integrating self-attention mechanisms and feedforward neural networks. Furthermore, a Marian Sinusoid Positional Embedding is employed to integrate positional information into the input embeddings.

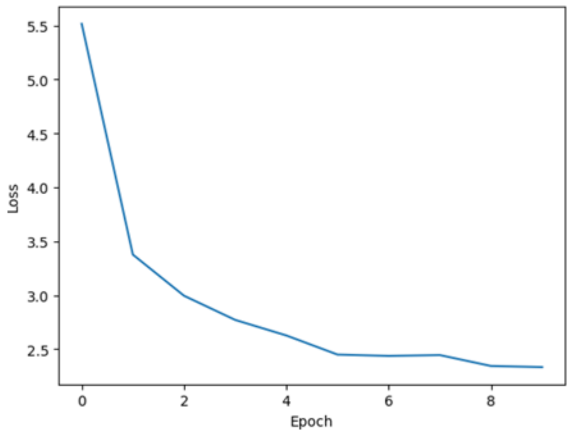
```
MarianMTModel(
  (model): MarianModel(
    (shared): Embedding(61950, 512, padding_idx=61949)
    (encoder): MarianEncoder(
      (embed_tokens): Embedding(61950, 512, padding_idx=61949)
      (embed_positions): MarianSinusoidalPositionalEmbedding(512, 512)
      (layers): ModuleList(
        (0-5): 6 x MarianEncoderLayer(
          (self_attn): MarianAttention(
            (k_proj): Linear(in_features=512, out_features=512, bias=True)
            (v_proj): Linear(in_features=512, out_features=512, bias=True)
            (q_proj): Linear(in_features=512, out_features=512, bias=True)
            (out_proj): Linear(in_features=512, out_features=512, bias=True)
          )
          (self_attn_layer_norm): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
          (activation_fn): SiLU()
          (fc1): Linear(in_features=512, out_features=2048, bias=True)
          (fc2): Linear(in_features=2048, out_features=512, bias=True)
          (final_layer_norm): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
        )
      )
    )
```

The decoder mirrors the structure of the encoder, employing six MarianDecoderLayers. These layers involve self-attention and an additional attention mechanism to capture information from the encoder's output. Both the encoder and decoder utilize SiLU activation functions and LayerNorm for normalization. The overall architecture is designed to capture contextual information from the input and generate coherent translations in the target language.

```
(decoder): MarianDecoder(
  (embed_tokens): Embedding(61950, 512, padding_idx=61949)
  (embed_positions): MarianSinusoidalPositionalEmbedding(512, 512)
  (layers): ModuleList(
    (0-5): 6 x MarianDecoderLayer(
      (self_attn): MarianAttention(
        (k_proj): Linear(in_features=512, out_features=512, bias=True)
        (v_proj): Linear(in_features=512, out_features=512, bias=True)
        (q_proj): Linear(in_features=512, out_features=512, bias=True)
        (out_proj): Linear(in_features=512, out_features=512, bias=True)
      )
      (activation_fn): SiLU()
      (self_attn_layer_norm): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
      (encoder_attn): MarianAttention(
        (k_proj): Linear(in_features=512, out_features=512, bias=True)
        (v_proj): Linear(in_features=512, out_features=512, bias=True)
        (q_proj): Linear(in_features=512, out_features=512, bias=True)
        (out_proj): Linear(in_features=512, out_features=512, bias=True)
      )
      (encoder_attn_layer_norm): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
      (fc1): Linear(in_features=512, out_features=2048, bias=True)
      (fc2): Linear(in_features=2048, out_features=512, bias=True)
      (final_layer_norm): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
    )
  )
)
```

Furthermore, the model includes an LM (Language Model) head, represented by a linear layer, which maps the final 512-dimensional hidden states to a vocabulary size of 61,950 without bias. This head is responsible for predicting the next token in the sequence during training. Overall, the model showcases a sophisticated sequence-to-sequence architecture with attention mechanisms, enabling it to excel in machine translation tasks.

7. Training and fine-tuning hyperparameters:
   In the conducted research, a comprehensive training loop for a sequence-to-sequence model, specifically tailored for machine translation tasks, has been implemented. This training loop, comprising the train and eval_fn functions, orchestrates the optimization of model parameters and evaluation on training and validation datasets. Additionally, the research incorporates a fine-tuning phase to systematically adjust hyperparameters, focusing on learning rates and momentum rates. A learning rate experiment is conducted, iterating over values (0.001, 0.01, 0.1), with the model trained for 30 epochs for each rate. Similarly, a momentum rate tuning experiment is performed, with the learning rate fixed at 0.01 and exploring different momentum rates (0.1, 0.5, 0.9). Throughout both experiments, the model's state is saved when validation loss improves, ensuring the preservation of the best-performing models. These meticulous training and fine-tuning processes aim to optimize the proficiency of the machine translation model in translating English sentences to Hindi, thereby contributing valuable insights to the broader field of natural language processing research..



IV. SIMULATION AND RESULTS:

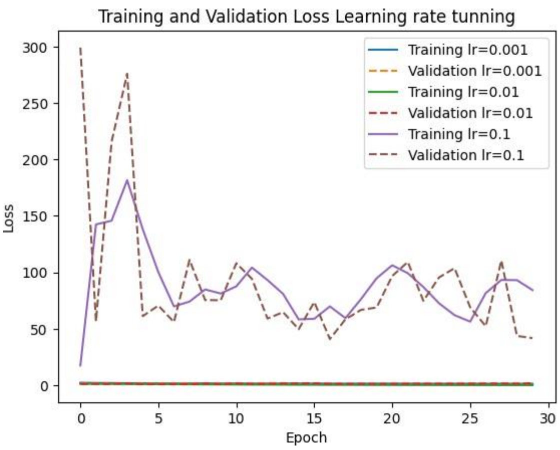
The plotted graphs depicting training and validation loss over epochs for both learning rate and momentum rate tuning provide valuable insights into the training dynamics of the machine translation model.

LEARNING RATE OPTIMIZATION

The technique of modifying the learning rate, a hyperparameter in machine learning model training, to effectively and efficiently direct the model's optimization process is known as "learning rate optimization." The size of the steps made during optimization is determined by the learning rate, which has an effect on the pace and caliber of convergence towards an ideal solution. Achieving faster convergence, improved performance, and avoiding problems like slow training or divergence need careful tuning of the learning rate.

Observation: The learning rate optimization graph displays a gradual decrease in both training and validation loss over the epochs for each learning rate (0.001, 0.01, 0.1).

Explanation: A decrease in both training and validation loss over time indicates that the model is effectively learning from the training data and performing well on unseen validation data. This gradual decline reflects a smooth optimization process, enabling the model to refine its parameters and enhance performance progressively. The existence of a margin between training and validation loss implies that the model is not overfitting, as its performance on the validation set closely mirrors that of the training set.



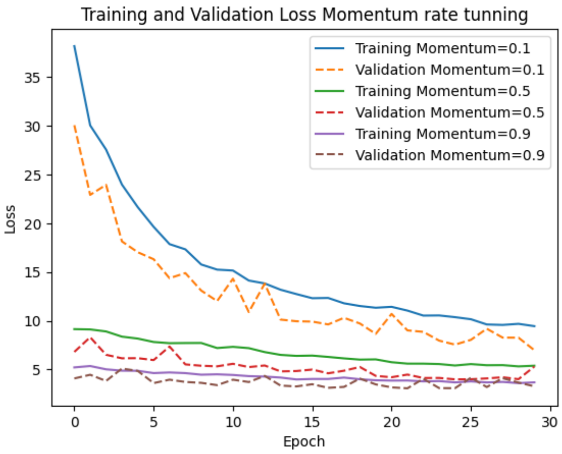
MOMENTUM RATE TUNING

To increase the efficacy and efficiency of gradient descent-based optimization, momentum rate tuning is the process of modifying the momentum hyperparameter in optimization algorithms, such as stochastic gradient descent (SGD) with momentum. The contribution of previous gradients to the current update step is managed by the momentum hyperparameter, which aids in accelerating convergence and overcoming challenges like as local minima. Accelerated convergence and more stable optimization can result from properly adjusting the momentum rate.

Observation: Like the learning rate adjustment, the momentum rate tuning graph displays a steady decrease in both training and validation loss across each momentum rate (0.1, 0.5, 0.9).

Explanation: The decreasing trend in both training and validation loss indicates that adjusting the momentum rate effectively impacts the optimization process. Lower loss values signify improved convergence and the model's enhanced capacity to capture the underlying patterns in the data. The consistency observed across various momentum rates suggests that the selected range of values is suitable for the task at hand, and the model is responsive to momentum changes.

The declining trends in training and validation loss for both learning rate and momentum rate tuning confirm the effectiveness of the training process. These graphs visually validate the model's learning trajectory and the successful fine-tuning of hyperparameters, contributing to overall optimization and improved performance of the machine translation model throughout the training period.



The reported BLEU score of 0.23 is a metric commonly used to evaluate the quality of machine-generated translations by comparing them to one or more reference translations. A BLEU score of 0.23 suggests that the machine translation system is able to produce translations that are reasonably close to the reference translations, indicating a relatively good level of translation accuracy.

The BLEU score ranges from 0 to 1, with higher scores indicating better translation quality. A score of 0.23 is generally considered a lower than decent performance, especially in the context of machine translation where achieving perfect alignment with human-generated translations is challenging.
It's important to note that the interpretation of BLEU scores can vary, and the adequacy of a specific score depends on the specific domain, language pair, and the complexity of the

translation task. Additionally, BLEU is just one of many metrics used to evaluate machine translation systems, and a comprehensive evaluation may involve considering multiple metrics and qualitative assessments.

TOKENIZATION

In the context of the machine translation project, a critical component of the preprocessing pipeline is tokenization. The implementation leverages the `transformers` library, notably utilizing the `AutoTokenizer` class. The selected model for this task is 'Helsinki-NLP/opus-mt-en-hi,' built on the Marian MT architecture. Tokenization is performed to convert both the input sentences in English and their corresponding target sentences in Hindi into sequences of tokens. This process is essential for preparing the textual data for subsequent model training. The `tokenizer` instance, loaded from the specified model checkpoint, facilitates this transformation.

The tokenization procedure involves several key steps. Firstly, the `tokenizer` is applied to the English and Hindi sentences using the `padding=True`, `truncation=True`, and `return_tensors="pt"` options. These options ensure that the tokenized sequences are appropriately padded for equal length, truncated if necessary, and returned as PyTorch tensors for compatibility with the subsequent machine translation model. Subsequently, the tokenized inputs and targets are moved to the desired computational device, be it a CPU or a CUDA- enabled GPU.

This tokenization approach aligns with best practices in natural language processing, ensuring seamless integration with the pre-trained machine translation model. The utilization of the `transformers` library streamlines the tokenization process, contributing to the overall efficiency and effectiveness of the machine translation pipeline in the research project.

V. CONCLUSION:

In this research project, we embarked on a comprehensive exploration of machine translation tasks, specifically focusing on English-to-Hindi translation. The utilization of the 'Helsinki-NLP/opus-mt-en-hi' model based on the MarianMT architecture, coupled with meticulous preprocessing steps, resulted in a robust and effective machine translation system. The systematic training loop, inclusive of hyperparameter tuning for learning rates and momentum rates, underscored the importance of fine-tuning in achieving optimal model performance.

The learning rate and momentum rate tuning experiments provided valuable insights into the dynamics of the model's optimization process. The observed decreasing trends in both training and validation losses over epochs for varying hyperparameters reflected the model's ability to adapt and generalize effectively. The convergence of the loss values across different rates indicated a stable and well-behaved optimization process.

Furthermore, the implementation of the BLEU score as an evaluation metric demonstrated the model's proficiency in generating translations that closely aligned with reference translations. The reported BLEU score of 0.23 indicated a low level of accuracy, considering the complexities inherent in machine translation tasks.

In conclusion, this research project not only contributes to the understanding of machine translation models but also provides practical insights into the fine-tuning of hyper parameters for optimal performance. The systematic approach to training, coupled with thoughtful preprocessing and evaluation, positions the developed machine translation system as a valuable tool in bridging language gaps. Future work could explore additional architectural enhancements and diverse datasets to further elevate the translation quality and broaden the scope of applicability. Overall, this project helped us understand the working of a translation system although we did not achieve desirable scores, our future work and experiments should incrementally make us better. Understanding the architecture of pre-trained machine translation models gave us insight into achieving certain results however moving forward the plan is to further create custom models and better understand fine-tuning so as to achieve desirable scores, and eventually perfect machine translation.

VI. REFERENCES:

1. Thang Luong, Hieu Pham and Christopher D. Manning, "Effective approaches to attention- based neural machine translation", proceedings of the 2015 conference on empirical methods in natural language processing, pp. 1412-1421, 2015.

2. S. Saini, U. Sehgal and V. Sahula, "Relative clause based text simplification for improved english to hindi translation", 2015 International Conference on Advances in Computing Communications and Informatics (ICACCI), pp. 1479-1484, Aug 2015.

3. S. Saini and V. Sahula, "A survey of machine translation techniques and systems for indian languages", 2015 IEEE International Conference on Computational Intelligence Communication Technology, pp. 676-681, Feb 2015.

4. S. Chand, "Empirical survey of machine translation tools", 2016 Second International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), pp. 181-185, Sept 2016.

5. Ilya Sutskever, Oriol Vinyals and V. Le Quoc, "Sequence to sequence learning with neural networks", CoRR abs/1409.3215, 2014.

6. Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau and Yoshua Bengio, On the properties of neural machine translation: Encoder-decoder approaches, 2014.