Neural Machine Translation Language English to Vietnamese

Trinh Duy Phong SID No. 20220065
Email: phong.td220065@hust.edu.vn
Nguyen Viet Anh SID No. 20225434
Email: anh.nv5434@hust.edu.vn
Nguyen Thanh Minh SID No. 20225450
Email: minh.nt2254504@hust.edu.vn
Hoang Trung Khai SID No. 20225502
Email: khai.ht225502@hust.edu.vn
Luu Thien Viet Cuong SID No. 20225477
Email: cuong.ltv225477@hust.edu.vn

HUST-SoICT

Abstract

This paper conducts a comprehensive comparative analysis of four prominent neural machine translation (NMT) models: Recurrent Neural Network (RNN) with and without attention mechanisms (including LSTM and GRU variants), Text-to-Text Transfer Transformer (T5), and a fine-tuned model utilizing BERT for encoding and BARTpho for decoding. The primary objective is to evaluate and benchmark the translation performance of these models across multiple language pairs and varying linguistic complexities.

Our study involves training and fine-tuning each model on a diverse set of corpora, assessing translation quality through BLEU scores and human evaluations. The findings reveal nuanced differences in performance, guiding researchers in selecting suitable models for specific translation requirements.

1 Introduction

Machine translation (MT) has revolutionized the landscape of cross-lingual communication, enabling seamless and efficient translation between diverse languages. As the demand for high-quality machine translation systems continues to grow, there arises a need for a comprehensive understanding of the strengths and limitations of various NMT models. In this context, this paper undertakes a rigorous comparative analysis of five prominent NMT models: Recurrent Neural Network without Attention layers (LSTM), LSTM and GRU with Attention layers , Text-to-Text Transfer Transformer (T5), MarianMT, and a finetuned model utilizing Bartpho as decoder and Bert as encoder . NMT models have undergone significant advancements, transitioning from traditional architectures like RNN to state-of-the-art transformer-based models such as T5, BARTpho and BERT, MarianMT. Each model brings its unique approach to handling linguistic complexities, making it imperative to scrutinize their performance across diverse language pairs and under various translation scenarios. The motivation behind this study lies in the necessity to provide practitioners and researchers with a nuanced understanding of the capabilities and limitations of these popular NMT models. By conducting a head-to-head comparison, we aim to offer insights into the specific strengths of each model and guide decision-making in selecting the most suitable NMT approach for different translation contexts. Our investigation encompasses not only quantitative metrics such as BLEU scores but also qualitative assessments through human evaluations. This holistic approach ensures a well-rounded evaluation of the translation quality delivered by each model. Additionally, we explore the impact of architectural choices and training strategies on the models' overall performance, contributing valuable insights to the ongoing discourse in the field. As we delve into the intricacies of GRU, LSTM with and without Attention, T5, and BERTBARTPho, MarianMT this paper aims to contribute a comprehensive benchmark for the NMT community, aiding researchers and practitioners in navigating the diverse landscape of machine translation technologies. Through this comparative analysis, we seek to illuminate the path toward more informed choices in deploying NMT systems tailored to specific translation requirements and linguistic characteristics. Machine translation (MT) has revolutionized cross-lingual communication, enabling seamless translations between diverse languages. This paper compares four NMT models:

- LSTM without attention
- GRU and LSTM with attention
- Text-to-Text Transfer Transformer (T5)
- Fine-tuned model using BERT as encoder and BARTpho as decoder
- MarianMT

NMT advancements transition from traditional RNN architectures to state-of-the-art Transformer-based models. By comparing quantitative metrics (e.g., BLEU scores) and qualitative assessments, this study provides benchmarks for advancing MT technologies.

2 Data Processing

2.1 Dataset PhoMT

Vietnam has achieved rapid economic growth in the last two decades. It is now an attractive destination for trade and investment. Due to the language barrier, foreigners often rely on automatic machine translation (MT) systems to translate Vietnamese texts. PhoMT is a high-quality and large-scale Vietnamese-English parallel dataset of 3.02M sentence pairs for machine translation. The dataset construction process involves collecting parallel document pairs, preprocessing for cleaning and quality, aligning parallel sentences, and postprocessing to filter out duplicates and verify set quality. Not all English-Vietnamese pairs in OpenSubtitles have high-quality translations, resulting in the removal of 15 percentage of document pairs. In MediaWiki, some Vietnamese documents contain untranslated English paragraphs, identified and filtered using fastText. Additionally, reference sections and tables in some MediaWiki and Blogspot documents are removed. The final dataset comprises 3312 high-quality English-Vietnamese parallel document pairs.

2.2 Tokenizer

2.2.1 BARTpho-word

2.2.2 BERT

2.2.3 Pho Bert base

2.2.4 T5

2.2.5 Sentence Piece

2.2.6 Keras Tokenizer

3 Models

3.1 T5 Model for summarization

Architecture
The T5 models are pre-trained transformer architecture developed by Google Research. It is based on the concept of framing all NLP tasks as a text-to-text problem, meaning that both the input and the output are treated as sequences of text. This framework makes T5 versatile for a variety of NLP applications, including translation, text generation, and classification.

Summarization with T5
For summarization, the T5 model takes the text to be summarized as input, prepended with a task-specific prefix, such as "summarize:". This informs the model that the input corresponds to a summarization task. The output is a concise summary of the input text.

Training Setup
We fine-tuned the T5-small model using the following configuration:

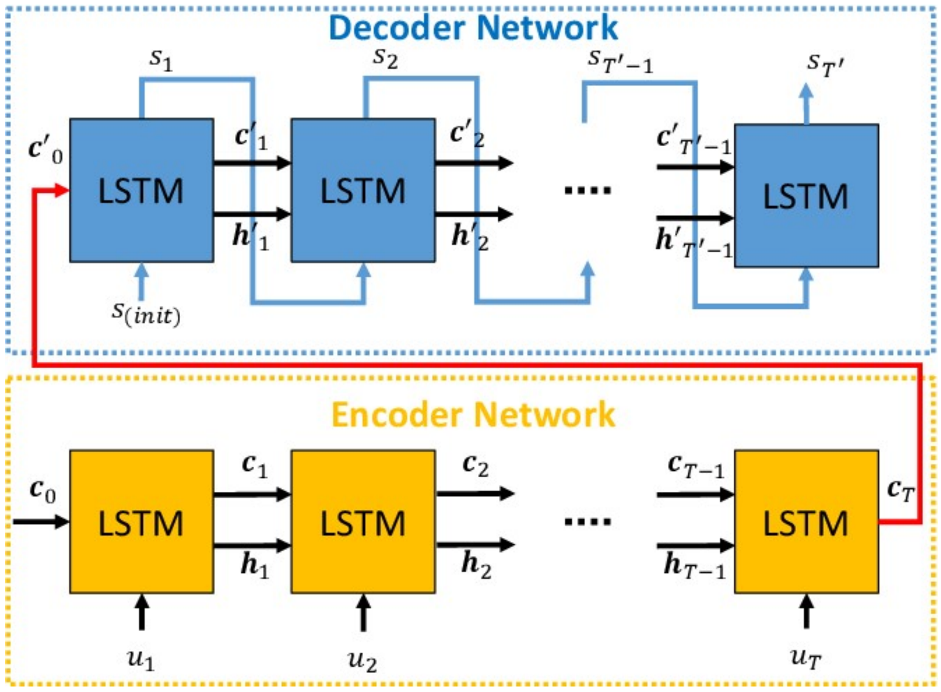
- Weight Decay: 0.01
- Learning Rate: 2e-5
- Batch Size: 8 for training, 8 for validation
- Epochs: 15 epochs for extended training
- Maximum Sequence Length: 64 tokens

- Checkpoints: Training resumes from the last checkpoint if available.
- Early Stopping: To prevent overfitting, we used early stopping with a patience of 3 epochs.

Objective:
The objective of our work is to fine-tune the T5-small model for summarizing long paragraphs or sentences into their shortest possible version. After obtaining the concise summary, we utilize a translation model to convert it into the final Vietnamese version.

3.2 Bidirectional-LSTM-based Encoder-Decoder without Attention

Architecture: The encoder-decoder network utilizes bidirectional LSTMs for the encoder. This architecture is designed to capture temporal dependencies in sequential data, processing the input sequence in both forward and backward directions to ensure that context from both past and future elements is incorporated. The key components are as follows:



- **Encoder:** The encoder consists of a stack of bidirectional LSTM layers that process the input sequence and encode it into a fixed-length context vector, summarizing the information from both directions.
- **Decoder:** The decoder, built with undirectional LSTM layers, generates the target sequence step-by-step using the context vector and previous outputs.

Parameters:

- **Encoder:**
  - input_dim = 27493
  - embedding_dim = 256
  - units = 128
  - input_length = 193
  - Drop_out = 0.2
  - Regularizer = 0.0001
- **Decoder:**
  - output_dim = 15289
  - embedding_dim = 256
  - units = 128
  - input_length = 232
  - Drop_out = 0.2
  - Regularizer = 0.0001

Dataset: For training and evaluation, we employ 2% of the PhoMT dataset:

- **Training Data:** 80% of the dataset
- **Validating Data:** 20% of the dataset

Preprocessing:

- **Text Cleaning:** A custom function `clean_text` is used to clean both English and Vietnamese sentences. Key steps include:

1. **Allowed Characters**: The regex retains lowercase letters, Vietnamese diacritics, and special characters like `'` `.`, and `,`.
2. **Whitespace Handling**: Consecutive whitespaces are replaced with a single space, and leading/trailing spaces are removed.

Tokenization:

- **English Tokenization:** The `Tokenizer` from Keras is used:
  - It tokenizes English sentences after cleaning.
  - It converts words into unique integer indices.
- **Vietnamese Tokenization**: The `Tokenizer` from Keras is used:
  - Wrapping each sentence with special tokens `<SOS>` (start of sentence) and `<EOS>` (end of sentence).
  - Using a custom tokenizer with `filters=''` and `oov_token='<UNK>'` to ensure no characters are filtered out and out-of-vocabulary words are mapped to `<UNK>`.
- **Sequence Conversion + Padding:**
  - Once tokenized, sentences are converted to sequences:
    - Each word/token is mapped to its corresponding integer index.
    - This step transforms text data into a numerical format suitable for input to deep learning models.
  - Sequences are padded to the maximum sequence length observed in the respective datasets:
    - Post-padding: Zeros are added at the end of sequences.
    - The `decoder_target_data` is created by removing the first token (`<SOS>`) from each sequence in the Vietnamese padded data.

Results: BLEU_1 score is 0.18, BLEU_4 is 0.027.

3.3 Bidirectional-LSTM with Attention

The encoder-decoder network utilizes bidirectional LSTMs for both the encoder and the decoder. This architecture is designed to capture temporal dependencies in sequential data, processing the input sequence in both forward and backward directions to ensure that context from both past and future elements is incorporated. The key components are outlined below:

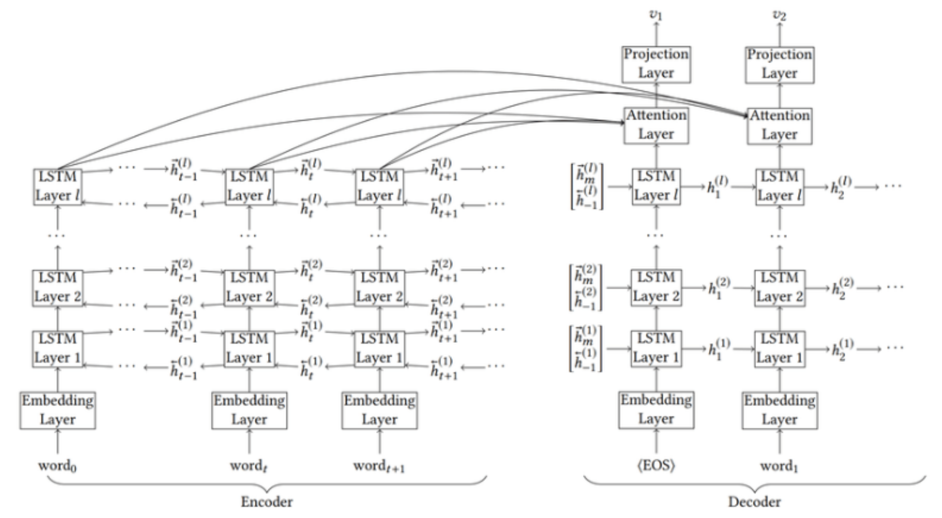


Figure 1: A Bidirectional Multilayer LSTM Encoder and Unidirectional LSTM Decoder (Source: Yin et al.)

- **Encoder:** The encoder consists of a stack of bidirectional LSTM layers that process the input sequence and encode it into a fixed-length context vector, summarizing the information from both directions.
- **Attention Mechanism:** During decoding, the attention mechanism computes a weighted alignment between the decoder's current state and the encoder's outputs, enabling the model to focus dynamically on relevant parts of the input sequence.
- **Decoder:** The decoder, built with unidirectional LSTM layers, generates the target sequence step-by-step using the context vector and previous outputs.

Parameters:

- `vocab_size = 64000`
- `units = 256`
- `input_length = 50`
- num_heads = 4
- `dropout = 0.2`

Dataset:

- **Raw Dataset:** The dataset consists of texts in two languages, English and Vietnamese. The English text serves as the context, while the Vietnamese text serves as the target. Each sentence pair is aligned by index, with the corresponding sentences stored in two separate files:
  - `"en"`: Represents the English translation.
  - `"vi"`: Represents the Vietnamese translation.
- **Preprocessing:** We use 2 pretrained models for tokenizing after cleaning data by allowing alphanumeric characters, spaces, and punctuation marks and Vietnamese characters
  1. `bert-base-uncased`: To archieve English token ids of context in original dataset
  2. **vinai/phobert-base:** To archieve Vietnamese token ids of target in original dataset

Results:

- BLEU-1 Score: `0.308`
- BLEU-4 Score: `0.07`
- Cosine similarity: `0.658`

Advantages:

- Effectively handles long-range dependencies with self-attention.
- Efficiently processes sequences in parallel for faster computations.

3.4 Bidirectional-GRU with Attention

Architecture: The model utilizes Gated Recurrent Units (GRUs) with an attention mechanism to process sequential data and improve the performance of multilingual tasks. GRUs are efficient recurrent neural network layers that reduce computational complexity compared to LSTMs, while still capturing temporal dependencies effectively. The attention mechanism enhances the model's ability to focus on relevant parts of the input sequence, especially for tasks involving long sequences or translations.

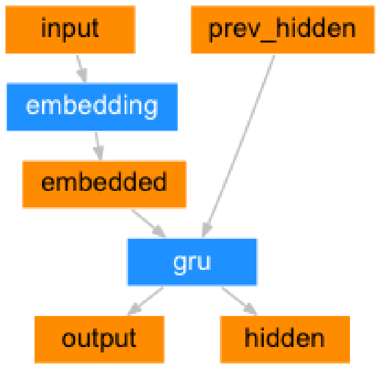


Figure 2: Encoder Structure

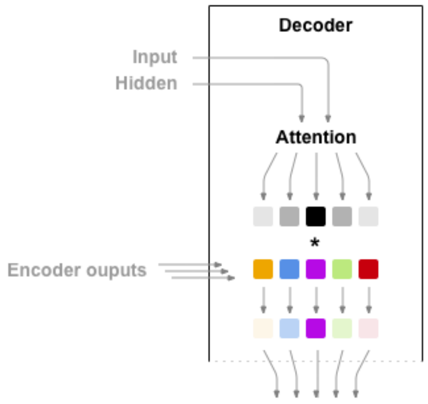


Figure 3: Decoder Structure

Key components of the architecture include:

- **Encoder:** A stack of GRU layers processes the input sequence and encodes it into a fixed-length context vector.
- **Attention Mechanism:** During decoding, the attention mechanism computes a weighted alignment between the decoder's current state and the encoder's outputs, enabling the model to focus dynamically on relevant parts of the input sequence.
- **Decoder:** The decoder GRU generates the target sequence, leveraging the context vector and the attention weights at each timestep to improve the accuracy of predictions.

Dataset:

- **Raw Dataset:** The dataset consists of texts in two languages, English and Vietnamese. The English text serves as the context, while the Vietnamese text serves as the target. Each sentence pair is aligned by index, with the corresponding sentences stored in two separate files:
  - `"en"`: Represents the English translation.
  - `"vi"`: Represents the Vietnamese translation.
- **Preprocessing:** We use 2 pretrained models for tokenizing after cleaning data by allowing alphanumeric characters, spaces, and punctuation marks and Vietnamese characters
  1. `bert-base-uncased:` To archieve English token ids of context in original dataset
  2. **vinai/phobert-base:** To archieve Vietnamese token ids of target in original dataset

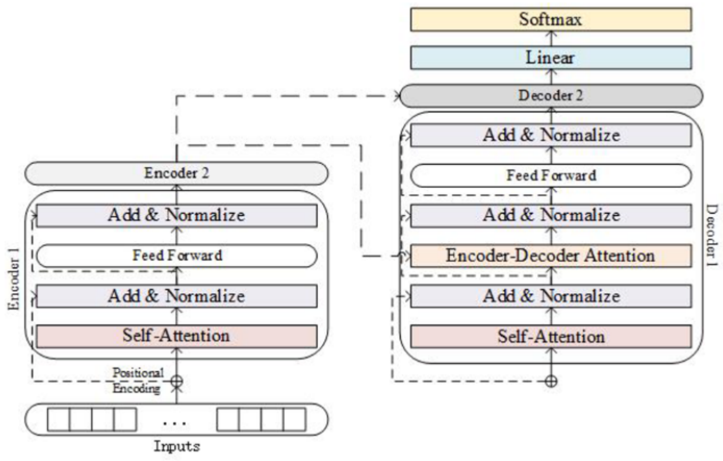
Results:

- BLEU-1 Score: `0.2`
- BLEU-4 Score: `0.03`
- Cosine similarity: `0.586442585289478`

3.5 T5 Model

Architecture
The T5 models are pre-trained transformer architecture developed by Google Research. It is based on the concept of framing all NLP tasks as a text-to-text problem, meaning that both the input and the output are treated as sequences of text. This framework makes T5 versatile for a variety of NLP applications, including translation, text generation, and classification.

T5 uses an encoder-decoder architecture, and it has been trained in multiple sizes, from small models (like t5-small) to large models (t5-11b), allowing flexibility based on computational resources. In this project, we fine-tuned the *T5-small* model on an English-Vietnamese parallel corpus.



Dataset
To train the model, we used a dataset of English-Vietnamese sentence pairs. The original dataset is 10 percent of the total PhoMT dataset with nearly 300 thousand examples. We also cut out the dataset size in stages to further improve model performance. The fine-tuning process was conducted in three phases:

- Phase 1: Trained on 20,000 sentence pairs, 4000 sentences for validation
- Phase 2: Trained on 80,000 sentence pairs, 8000 sentences for validation
- Phase 3: Trained on 200,000 sentence pairs, 8000 sentences for validation

This progressive training approach enabled the model to generalize better as it encountered more diverse data during each training phase.

- Training Set (24.000): Used to fine-tune the model.
- Validation Set (12000): Used to evaluate the model during training.

Tokenization
Before fine-tuning the model, the text data needs to be tokenized. We used two tokenizers:

- en_tokenizer (T5Tokenizer): A pre-trained T5Tokenizer from the Hugging Face library, which tokenizes English text and converts it into a sequence of tokens that can be fed into the model.
- vi_tokenizer (SentencePiece): A custom tokenizer trained using the SentencePiece algorithm to handle Vietnamese text. I trained this tokenizer with the same configuration as the T5Tokenizer, including the vocabulary size (32128) and other parameters, to ensure full compatibility with the T5 model.

Training Setup
We fine-tuned the T5-small model using the following configuration:

- Warmup Steps: 500
- Learning Rate: 5e-4
- Batch Size: 128 for training, 64 for validation
- Epochs: 5 epochs for extended training
- Maximum Sequence Length: 64 tokens
- Checkpoints: Training resumes from the last checkpoint if available.
- Early Stopping: To prevent overfitting, we used early stopping with a patience of 3 epochs.
- Number of Workers: 4 data loader workers to speed up data loading.
- Weight Decay: 0.1, to prevent overfitting
- Evaluation Metric: Validation loss

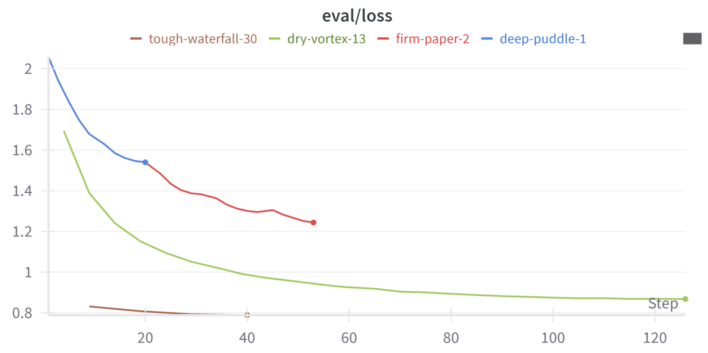
We employed the EarlyStoppingCallback to monitor validation loss and stop training if no improvement was observed after 2 consecutive epochs.

Training Process
The fine-tuning process was carried out in three stages to enhance the model's generalization and improve performance:

- Phase 1 - Initial training phase with 20,000 sentence pairs.
  During this stage, the model learns the basic structure of English-Vietnamese translation and acquires fundamental translation patterns to allow the model to understand simple sentence structures and produce basic translations.

- Phase 2 - Trained on a larger set of 100,000 sentence pairs.
  This enables the model to learn more different sentence structures, especially for longer and more complex sentences.

- Phase 3 – Training with 200,000 Training Pairs
  In this phase, the model is exposed to 200,000 sentence pairs, covering a wider range of vocabulary and sentence patterns. This improce the coverage of translation scenarios, leading to better generalization and also handle rare words better.



Results
Phase 1: Blue line
The initial loss was high but gradually decreased over the steps then stabilized around 1.5-1.6. The training went smoothly, but the loss was still quite high compared to the target threshold. This phase served as the starting point, helping the model learn basic sentence structures but without achieving high accuracy.

Phase 2: Red line
The loss started around 1.6 and decreased gradually. The loss dropped to 1.2-1.3 at the end. This phase allowed the model to learn more complex structures due to the increased data size, leading to a noticeable drop in loss.

Phase 3: Green line
This phase had the best result with the lowest loss. In this phase, the model made good use of the large dataset and generalized well, achieving the goal of reducing the validation loss to under 1.0.

After training the T5-small model, we evaluated its performance on a held-out validation set using the BLEU score. The BLEU-1 score is and BLEU - 4 score is...

3.6 Finetuned model with Bert and Bartpho-Word as encoder-decoder

Architecture: Pre-trained Transformer for multilingual tasks. The effectiveness of initializing sequence-to-sequence models with pretrained checkpoints for sequence generation tasks was shown in [**?**]. In this approach, Bert serves as the encoder and Bartpho-Word as the decoder. Bert, designed for bidirectional language representation, excels in extracting contextual embeddings from input sequences. Meanwhile, Bartpho-Word, adapted for Vietnamese tasks, extends Bart's capabilities with a specific focus on word-level tokenization and Vietnamese language nuances. This architecture combines Bert's robust understanding of input texts with Bartpho-Word's generative strengths, enabling efficient handling of sequence-to-sequence tasks in multilingual contexts, particularly Vietnamese.

Training Configuration: The model is fine-tuned on task-specific data using a sequence-to-sequence learning objective. During training:

- The encoder takes in tokenized input sequences, encoded using the Bert tokenizer. These are padded to a maximum sequence length to ensure batch consistency.

- The decoder generates outputs token by token, leveraging Bartpho-Word's word-level tokenization to produce grammatically accurate and semantically coherent outputs.

- Cross-entropy loss is used as the primary objective, with teacher forcing applied during training to guide the decoder.

- For the decoder's start token configuration, the cls_token_id of BartPho is used as the start token, and the padding token of BartPho is applied consistently as the pad token for both encoder and decoder structures.

- Learning Rate: 5e-5
- Batch Size: 64 for training, 64 for validation
- Epochs: 5 epochs for extended training
- Maximum Sequence Length: 64 tokens
- Early Stopping: To prevent overfitting, we used early stopping with a patience of 2 epochs.
- Weight Decay: 0.1, to prevent overfitting
- Evaluation Metric: Rouge

Results: Initially, the model achieved very low scores, with BLEU_1: 0.03 and approximately 0 for BLEU_4. To fine-tune this model, we used 297,799 data samples (10 percent of the total PhoMT dataset) and trained it for 10 epochs with early stopping. After fine-tuning, the scores improved significantly to 0.56 and 0.30 for BLEU_1 and BLEU_4, respectively.

3.7 MarianMT

Architecture: Marian is a sequence-to-sequence model that is specifically designed for machine translation. It uses an encoder-decoder architecture with attention mechanisms to generate translations from one language to another. Marian is unique in that it can be trained on multiple languages simultaneously, making it a versatile model for multilingual translation tasks.
Training: MarianMT is trained using a diverse and extensive dataset of parallel texts, which consist of sentence pairs in source and target languages. The training process incorporates tokenization and subword units (such as byte-pair encoding) to handle rare and unseen words effectively. This ensures that the model learns to translate accurately across a wide range of language pairs, including low-resource languages.
Training Setup
We fine-tuned the MarianMT model using the following configuration:

- Learning Rate: 5e-5
- Batch Size: 64 for training, 64 for validation
- Epochs: 5 epochs for extended training
- Maximum Sequence Length: 64 tokens
- Checkpoints: Training resumes from the last checkpoint if available.
- Early Stopping: To prevent overfitting, we used early stopping with a patience of 2 epochs.

Training results: For this task, we use the MarianMT model: "Helsinki-NLP/opus-mt-en-vi" (a model trained for the task of translating English to Vietnamese) trained for 5 epochs with 297,799 data samples (which is 10 percent of the total PhoMT dataset). Before fine-tuning, MarianMT achieved BLEU_1 and BLEU_4 scores of 0.39 and 0.16, respectively. After fine-tuning, the scores improved to 0.61 and 0.33, the highest among the compared methods.

4 Evaluation Metric and Results

4.1 Evaluation Metric

4.1.1 BLEU metric

BLEU metric: BLEU metric: BLEU (BiLingual Evaluation Understudy) is a metric for automatically evaluating machine-translated text. The BLEU score is a number between zero and one that measures the similarity of the machine-translated text to a set of high quality reference translations.

BLEU formula:

$$\text{BLEU} = \min\left(1, \exp\left(1 - \frac{\text{reference-length}}{\text{output-length}}\right)\right)\left(\prod_{i=1}^{4} \text{precision}_i\right)^{1/4}$$

Where:

$$\text{precision}_i = \frac{\sum_{\text{snt}\in\text{Cand-Corpus}} \sum_{i\in\text{snt}} \min\left(m_i^{\text{cand}}, m_i^{\text{ref}}\right)}{\sum_{\text{snt}\in\text{Cand-Corpus}} \sum_{i\in\text{snt}} m_i^{\text{cand}}}$$

$$w_i^t = \frac{\sum_{\text{snt}\in\text{Cand-Corpus}} \sum_{i\in\text{snt}} m_i^{\text{cand}}}{\sum_{\text{snt}'\in\text{Cand-Corpus}} \sum_{i'\in\text{snt}'} m_i^{\text{cand}}}$$

A value of 0 means that the machine-translated output has no overlap with the reference translation (which indicates a lower quality) while a value of 1 means there is perfect overlap with the reference translations (which indicates a higher quality).

BLEU_N: In the BLEU metric, BLEU_N refers to the evaluation based on n-grams of length N, where an n-gram is a contiguous sequence of N words in a sentence. BLEU_N calculates how well the machine-translated text matches the reference translations by comparing n-grams of size N from both texts.
Range of BLEU_N:

- BLEU-1: Uses unigrams (1-grams) to assess word-level matches.
- BLEU-2: Considers bigrams (2-grams) to measure contextual and syntactic correctness.
- BLEU-3: Evaluates trigrams (3-grams) for more extended phrase consistency.
- BLEU-4: Uses 4-grams, capturing higher-level coherence.

4.1.2 Cosin Similarity

Definition: Cosine Similarity is a metric that measures the similarity between two non-zero vectors in a multidimensional space. It calculates the cosine of the angle between the vectors, which indicates their directional similarity. The value ranges from -1 to 1, where:

- 1: Vectors are identical in direction.
- 0: Vectors are orthogonal (no similarity).
- -1: Vectors are opposite in direction.

Cosine similarity formula:

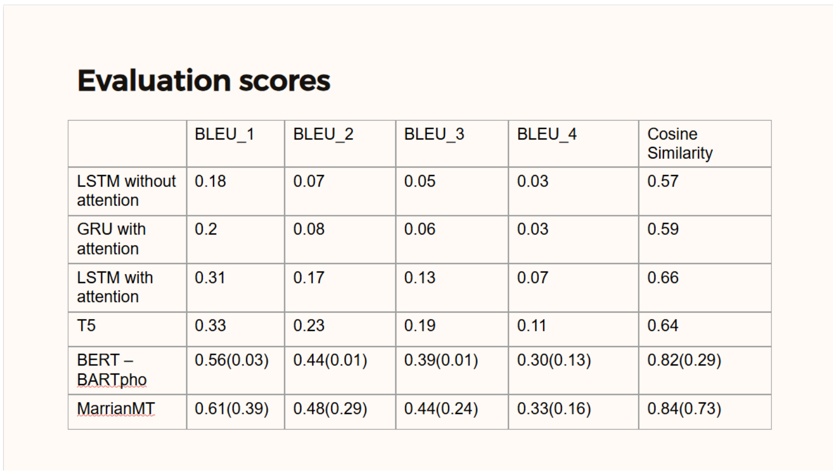
$$\text{Cosine Similarity} = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|}$$

Where:

- $\mathbf{A} \cdot \mathbf{B}$ is the dot product of vectors $\mathbf{A}$ and $\mathbf{B}$.
- $\|\mathbf{A}\|$ and $\|\mathbf{B}\|$ are the magnitudes (norms) of $\mathbf{A}$ and $\mathbf{B}$.

4.2 Evaluation Results

The evaluation results are summarized in the figure below:



Evaluation scores

	BLEU_1	BLEU_2	BLEU_3	BLEU_4	Cosine Similarity
LSTM without attention	0.18	0.07	0.05	0.03	0.57
GRU with attention	0.2	0.08	0.06	0.03	0.59
LSTM with attention	0.31	0.17	0.13	0.07	0.66
T5	0.33	0.23	0.19	0.11	0.64
BERT – BARTpho	0.56(0.03)	0.44(0.01)	0.39(0.01)	0.30(0.13)	0.82(0.29)
MarrianMT	0.61(0.39)	0.48(0.29)	0.44(0.24)	0.33(0.16)	0.84(0.73)

Figure 4: Evaluation results showing the performance metrics of the models.

5 References

- ViT5: Pretrained Text-to-Text Transformer for Vietnamese Language Generation
- MTet: Multi-domain Translation for English and Vietnamese
- PhoMT: A High-Quality and Large-Scale Benchmark Dataset for Vietnamese-English Machine Translation