# English - Vietnamese Translation Model

Doan Manh Tan[1] and Tran Thi Kim Huynh[2]

[1]21120326@student.hcmus.edu.vn
[2]21120607@student.hcmus.edu.vn

June 2024

## 1 Introduction

This report details the use of the Helsinki-NLP/opus-mt-en-vi[2] model for English to Vietnamese translation. The model is fine-tuned on the PhoMT dataset using the MarianMT[3] framework, a fast neural machine translation framework. This report covers the pre-trained model's architecture, training dataset, performance metrics, tokenization method, and the specifics of the PhoMT dataset and the fine-tuning process.

## 2 Pre-trained Model: Helsinki/opus-mt-en-vi

### 2.1 Overview

The Helsinki-NLP/opus-mt-en-vi model is part of the OPUS-MT project, which provides machine translation models for a variety of language pairs. This specific model is designed to translate from English to Vietnamese.

### 2.2 Training Dataset

The Helsinki-NLP/opus-mt-en-vi model was trained on the OPUS dataset, a large collection of parallel corpora compiled from various sources. The OPUS dataset includes text from sources like TED talks, EU legislation, and other multilingual texts, providing a rich resource for training translation models.

### 2.3 Training Framework: Marian

The model is trained using Marian, a highly efficient neural machine translation framework. Marian supports both training and inferencing and is optimized for speed and memory efficiency, making it suitable for large-scale translation tasks.

### 2.4 Performance Metrics

The performance of the Helsinki-NLP/opus-mt-en-vi model is evaluated using the BLEU (Bilingual Evaluation Understudy) score, a standard metric for assessing the quality of machine-translated text. Higher BLEU scores indicate better translation quality. The Helsinki-NLP/opus-mt-en-vi model achieves a BLEU score that is competitive with other state-of-the-art translation models, though specific scores can vary depending on the evaluation dataset.

## 3 Model Architecture: MarianMTModel

The MarianMTModel is based on the transformer architecture introduced by Vaswani et al. in their 2017 paper "Attention is All You Need". The key components of the transformer architecture include:

- **Encoder-Decoder Structure**: The model consists of an encoder to process the input text and a decoder to generate the translated output.
- **Self-Attention Mechanism**: Both encoder and decoder utilize self-attention to weigh the importance of different words in the input sequence.
- **Multi-Head Attention**: Allows the model to focus on different parts of the input text simultaneously.
- **Positional Encoding**: Adds information about the position of words in the sequence, which is crucial since transformers do not inherently understand word order.
- **Feed-Forward Networks**: After the attention layers, feed-forward neural networks are used for further processing.

### 3.1 Training Process

The MarianMTModel is trained using parallel corpora, with the objective of minimizing the cross-entropy loss between the predicted and true token distributions. Techniques like teacher forcing are employed to improve convergence and translation quality.

## 4 Tokenization: MarianTokenizer

### 4.1 SentencePiece Model

The MarianTokenizer utilizes the SentencePiece[4] library for text tokenization. SentencePiece implements two subword segmentation algorithms, byte-pair encoding (BPE)[6] and unigram language model[5]. This framework treats text as a sequence of Unicode characters and uses subword units for tokenization. Therefore, there is no language-dependent logic.

### 4.2 Byte-Pair Encoding (BPE)

- **Subword Segmentation**: BPE[6] iteratively merges the most frequent pairs of characters or bytes to create subword units.
- **Vocabulary Mapping**: Each subword is mapped to a unique integer ID, forming the basis for model input.

### 4.3 Unigram Language Model

In addition to BPE, SentencePiece implements the Unigram Language Model, which is a probabilistic model used for subword segmentation[5]. The Unigram Language Model selects the most probable subword sequence based on the likelihood of subword units.

### 4.4 Tokenization Process

- **Text Normalization**: The text is normalized to ensure consistency.
- **Subword Tokenization**: The text is segmented into subwords using BPE.
- **Encoding**: The sequence of subword IDs is padded or truncated to a fixed length and converted into tensors.

### 4.5 Decoding Process

- **Detokenization**: The output sequence of subword IDs is mapped back to subwords.
- **Subword Merging**: The subwords are merged to reconstruct the original text as closely as possible.

## 5 Fine-Tuning on PhoMT Dataset

### 5.1 Dataset Overview

The PhoMT[1] dataset is a bilingual corpus specifically designed for English to Vietnamese translation tasks. It includes parallel texts in English and Vietnamese, split into tokenization and detokenization. Each of the folders is splitted into training (2977999 examples), dev (18719 examples), and test sets (19151 examples). We utilized the data in the tokenization folder.

### 5.2 Training Process

The fine-tuning process involves the following steps:

1. **Data Preparation**: The PhoMT dataset is loaded and preprocessed. Texts are tokenized using the MarianTokenizer.
2. **Model Fine-Tuning**: The pre-trained MarianMTModel is fine-tuned on the PhoMT training data. The training process optimizes the model parameters to better capture the nuances of English to Vietnamese translation specific to the PhoMT dataset.
3. **Evaluation**: The fine-tuned model is evaluated on the validation set using BLEU scores to ensure it is learning effectively. Adjustments are made as necessary. Finally, model is evaluated on the test set to capture the overall performance.

### 5.3 Training Arguments

During fine-tuning, the following training arguments were used to control various aspects of the training process:

- **Learning Rate**: A learning rate of 2e-5 was used to balance the speed of convergence with stability.
- **Batch Size**: The batch size was set to 8 to fit within the memory constraints of the available GPU resources.
- **Epochs**: The model was trained for 3 epochs to ensure sufficient learning without overfitting.
- **Optimizer**: The AdamW optimizer was used, which is well-suited for training transformer models.
- **Weight Decay**: A weight decay of 0.1 was applied to regularize the model and prevent overfitting.
- **Evaluation Strategy**: The model was evaluated at the end of each epoch to monitor performance and adjust training as necessary.

### 5.4 Limitations in Training Resources

Due to the limitations in available GPU resources on Google Colab, each training session could only process up to 15,000 samples from the training set and evaluate up to 8,000 samples from the validation set. This constraint impacted the overall training time and required careful selection of training and evaluation batches to ensure meaningful progress.

### 5.5 Results

The fine-tuned model achieved a BLEU score of 35.26 on the PhoMT evaluation dataset and 34.81 on test dataset. This indicates that the model has successfully adapted to the specific characteristics of the PhoMT corpus, providing more accurate translations for this domain.

# 6 Conclusion

The Helsinki-NLP/opus-mt-en-vi model, built using the MarianMT framework, demonstrates robust performance in English to Vietnamese translation tasks. By fine-tuning this model on the PhoMT dataset, we achieve further improvements, showcasing the model's adaptability to specific translation domains. The combination of transformer architecture, efficient training processes, and advanced tokenization techniques like SentencePiece and BPE contribute to the success of this machine translation system.

# References


[1] Long Doan et al. "PhoMT: A High-Quality and Large-Scale Benchmark Dataset for Vietnamese-English Machine Translation". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021, pp. 4495–4503.

[2] Helsinki-NLP. *opus-mt-en-vi*. 2020. URL: `https://huggingface.co/Helsinki-NLP/opus-mt-en-vi`.

[3] Marcin Junczys-Dowmunt et al. *Marian: Fast Neural Machine Translation in C++*. 2018. URL: `https://marian-nmt.github.io/`.

[4] Taku Kudo. "SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 2018. URL: `https://www.aclweb.org/anthology/D18-2012`.

[5] Taku Kudo. "Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates". In: 2018. URL: `https://aclanthology.org/P18-1007/`.

[6] Rico Sennrich, Barry Haddow, and Alexandra Birch. "Neural Machine Translation of Rare Words with Subword Units". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2016. URL: `https://www.aclweb.org/anthology/P16-1162`.