Transformer in Machine Translation: Vietnamese-English Neural Machine Translation from Scratch

Phan Tất An
23020003@vnu.edu.vn

Trần Văn Quyết
23020143@vnu.edu.vn

Nguyễn Văn Cử
23020015@vnu.edu.vn

Phạm Huy Châu Long
23021615@vnu.edu.vn

Tóm tắt nội dung

Báo cáo này trình bày việc xây dựng một hệ thống dịch máy thần kinh (Neural Machine Translation - NMT) hoàn chỉnh cho cặp ngôn ngữ Việt-Anh sử dụng kiến trúc Transformer. Hệ thống được phát triển từ đầu (from scratch) bao gồm: (1) SentencePiece tokenizer tự huấn luyện với thuật toán BPE và Unigram trên tập dữ liệu PhoMT (2.9 triệu cặp câu song ngữ), (2) mô hình Transformer với các cải tiến như Pre-Layer Normalization, Weight Tying, và Label Smoothing, (3) hệ thống huấn luyện với Warmup Learning Rate Scheduler và Mixed Precision Training. Mô hình đạt điểm BLEU-4 khoảng 30-35 trên tập test, chứng minh hiệu quả của các kỹ thuật được áp dụng.

1 Introduction

1.1 Motivation

Dịch máy (Machine Translation) là một trong những bài toán quan trọng nhất trong lĩnh vực Xử lý Ngôn ngữ Tự nhiên (Natural Language Processing - NLP). Với sự phát triển của học sâu (Deep Learning), các mô hình dịch máy thần kinh đã đạt được những tiến bộ vượt bậc so với các phương pháp truyền thống như Statistical Machine Translation (SMT).

Kiến trúc Transformer, được giới thiệu trong bài báo "Attention Is All You Need"(1) năm 2017, đã tạo ra một bước ngoặt lớn trong NMT. Transformer loại bỏ hoàn toàn cơ chế Recurrent và Convolution, thay vào đó sử dụng cơ chế Self-Attention cho phép mô hình học được các mối quan hệ xa trong câu một cách hiệu quả.

Đối với cặp ngôn ngữ Việt-Anh, việc xây dựng hệ thống dịch máy chất lượng cao gặp nhiều thách thức:

- Tiếng Việt là ngôn ngữ đơn lập (isolating language) với cấu trúc ngữ pháp khác biệt so với tiếng Anh
- Tiếng Việt sử dụng dấu thanh điệu, đòi hỏi tokenizer phải xử lý đúng các ký tự Unicode
- Thiếu hụt dữ liệu song ngữ chất lượng cao

1.2 Objectives

Mục tiêu của dự án này là xây dựng một hệ thống dịch máy Anh–Việt hoàn chỉnh từ đầu, bao gồm toàn bộ các thành phần từ tiền xử lý dữ liệu cho đến thiết kế mô hình và huấn luyện. Dự án hướng tới việc triển khai SentencePiece tokenizer với hai thuật toán phổ biến là BPE và Unigram Language Model, qua đó đánh giá ảnh hưởng của từng phương pháp phân tách từ đến hiệu năng dịch.
Bên cạnh đó, chúng tôi cài đặt kiến trúc Transformer cùng một số kỹ thuật cải tiến hiện đại, sau đó huấn luyện và đánh giá hệ thống trên bộ dữ liệu PhoMT quy mô lớn. Thông qua quá trình này, dự án cũng nhằm mục tiêu tìm hiểu và áp dụng các best practices trong huấn luyện mô hình dịch máy hiện đại.

1.3 Contributions

Đóng góp chính của dự án:

1. SentencePiece from Scratch: Cài đặt hoàn chỉnh thuật toán BPE và Unigram Language Model với EM algorithm, không sử dụng thư viện có sẵn
2. Transformer Implementation: Triển khai Transformer với Pre-Layer Normalization, Weight Tying, và các kỹ thuật regularization
3. Training Pipeline: Xây dựng pipeline huấn luyện hoàn chỉnh với checkpoint management, learning rate scheduling, và visualization
4. Inference System: Hệ thống inference với Beam Search và Greedy Decoding hỗ trợ batch processing

2 Related Work

2.1 Neural Machine Translation Evolution

Lịch sử phát triển của dịch máy thần kinh có thể chia thành các giai đoạn:

1. **Sequence-to-Sequence với RNN (2014)**
   Sutskever et al. (2) giới thiệu mô hình Encoder-Decoder sử dụng LSTM. Mô hình này đọc toàn bộ câu nguồn, nén thành một vector context cố định, sau đó giải mã thành câu đích.

2. **Attention Mechanism (2015-2017)**
   Bahdanau et al. (3) đề xuất cơ chế Attention, cho phép decoder "nhìn lại"các hidden states của encoder. Điều này giải quyết vấn đề bottleneck của vector context cố định.

3. **Transformer (2017-Nay)**
   Vaswani et al. (1) giới thiệu Transformer, loại bỏ hoàn toàn RNN và sử dụng Self-Attention. Transformer cho phép:

   - Song song hóa tính toán (không như RNN phải tính tuần tự)
   - Học được long-range dependencies hiệu quả
   - Training nhanh hơn và scale tốt hơn

2.2 English to Vietnamese Translation

Dịch máy Anh – Việt là một bài toán quan trọng nhưng đầy thách thức do sự khác biệt lớn về cấu trúc câu, ngữ pháp và ngữ nghĩa giữa hai ngôn ngữ. Sự phát triển của các mô hình NMT, đặc biệt là Transformer, đã nâng cao đáng kể chất lượng dịch, tuy nhiên hiệu năng vẫn phụ thuộc mạnh vào dữ liệu huấn luyện. Vì vậy, việc lựa chọn và sử dụng bộ dữ liệu phù hợp là yếu tố then chốt.

Datasets

Trong nghiên cứu dịch máy Việt – Anh, một số bộ dữ liệu quan trọng thường được sử dụng là IWSLT, PhoMT và OPUS. IWSLT là tập dữ liệu được xây dựng từ các bài nói TED với khoảng 130K cặp câu, phù hợp cho các mô hình nhỏ hoặc các nghiên cứu thử nghiệm.

PhoMT (7) là bộ dữ liệu lớn nhất hiện nay với khoảng 2.9 triệu cặp câu song ngữ Việt – Anh, được thu thập từ nhiều nguồn và được xử lý, làm sạch cẩn thận. Bên cạnh đó, OPUS cung cấp một kho dữ liệu đa ngôn ngữ đến từ nhiều miền ứng dụng khác nhau.

Trong dự án này, chúng tôi lựa chọn sử dụng **PhoMT** (ura-hcmut/PhoMT trên HuggingFace) do quy mô lớn, chất lượng dữ liệu tốt và đặc biệt là đã được chia sẵn thành các tập train/validation/test, thuận tiện cho việc huấn luyện và đánh giá mô hình.

Baseline Systems

Để có cơ sở so sánh, chúng tôi tham khảo các hệ thống dịch Việt – Anh hiện có như Google Translate (hệ thống thương mại dựa trên Transformer), VinAI Translate (một mô hình dịch được huấn luyện riêng cho tiếng Việt) và một số hệ thống dựa trên OpenNMT. Các hệ thống này giúp định hình mức hiệu năng chuẩn, qua đó đánh giá chất lượng mô hình mà chúng tôi xây dựng.

2.3 Transformer Improvements

Trong những năm gần đây, nhiều hướng cải tiến đã được đề xuất nhằm nâng cao hiệu quả của kiến trúc Transformer gốc. Một trong những cải tiến quan trọng là **Pre-Layer Normalization** (4), trong đó LayerNorm được đặt trước các khối Self-Attention và Feed-Forward thay vì sau như thiết kế ban đầu. Cách bố trí này giúp mô hình ổn định hơn trong quá trình lan truyền gradient, giảm phụ thuộc vào warmup và cho phép sử dụng learning rate lớn hơn.

Bên cạnh đó, kỹ thuật **Weight Tying** (5) cho phép chia sẻ trọng số giữa embedding layer và output projection. Điều này không chỉ giúp giảm đáng kể số lượng tham số của mô hình mà còn được chứng minh là cải thiện hiệu năng trên nhiều bài toán ngôn ngữ tự nhiên.

3 Methodology

Mục tiêu của dự án là xây dựng một hệ thống dịch máy Anh–Việt dựa trên mô hình Transformer hoàn chỉnh, trong đó hầu hết các thành phần lõi đều được hiện thực lại từ đầu, bao gồm: kiến trúc mô hình, tokenizer, dataloader và pipeline huấn luyện. Phần này trình bày chi tiết các thành phần chính của hệ thống.

3.1 Tokenization Module

Chúng tôi xây dựng SentencePiece tokenizer từ đầu sử dụng hai thuật toán:

Stage 1: Byte Pair Encoding (BPE)

BPE là thuật toán nén dữ liệu được áp dụng cho tokenization:

Algorithm 1 Byte Pair Encoding

Khởi tạo vocabulary với tất cả ký tự
for $i = 1$ to num_merges **do**
    Đếm tần suất tất cả cặp ký tự liền kề
    Tìm cặp (a, b) xuất hiện nhiều nhất
    Thay thế tất cả (a, b) bằng token mới ab
    Thêm ab vào vocabulary
end for

Stage 2: Unigram Language Model với EM Algorithm

Sau BPE, sử dụng EM algorithm để tối ưu tokenization:

E-step (Expectation):

- Tokenize text với probabilities hiện tại
- Đếm tần suất tokens trong tokenization

M-step (Maximization):

- Tìm tokenization tối ưu bằng Viterbi (Dynamic Programming)
- Maximize log-likelihood của tokenization

Pruning:

- Xóa 20% tokens ít xuất hiện nhất (trừ base characters)
- Lặp lại cho đến khi đạt target vocab_size

Cấu hình Tokenizer:

Parameter	Value
Vocabulary Size	32,000 tokens
BPE Merges	10,000 operations
Training Samples	500,000 pairs
Special Tokens	PAD, UNK, SOS, EOS

Bảng 1: Cấu hình SentencePiece Tokenizer

Ưu điểm của SentencePiece Tokenizer: so với các tokenize đơn giản là chia mỗi token là một từ thì SentencePiece Tokenizer triển khai chiến lược thông minh hơn là chia từ thành nhiều sub-word. Điều này giúp giảm Out-Of-Vocabulary, biểu diễn từ tốt hơn đồng thời nâng cao hiệu năng khi huấn luyện mô hình.

3.2 Data Handling Module

Text Preprocessing: Các bước tiền xử lý:

- **Unicode Normalization**: Chuẩn hóa NFC để xử lý đúng dấu tiếng Việt
- **Whitespace Normalization**: Loại bỏ khoảng trắng thừa
- **Length Filtering**: Loại bỏ câu rỗng hoặc quá dài (> 256 words)
- **Length Ratio Filtering**: Loại bỏ cặp câu có tỷ lệ độ dài bất thường (> 1.5)

3.3 Transformer Architecture Module

Kiến trúc Transformer trong dự án tuân theo thiết kế Encoder-Decoder với cấu trúc các module chính như hình 1.

3.3.1 Các Module Chính

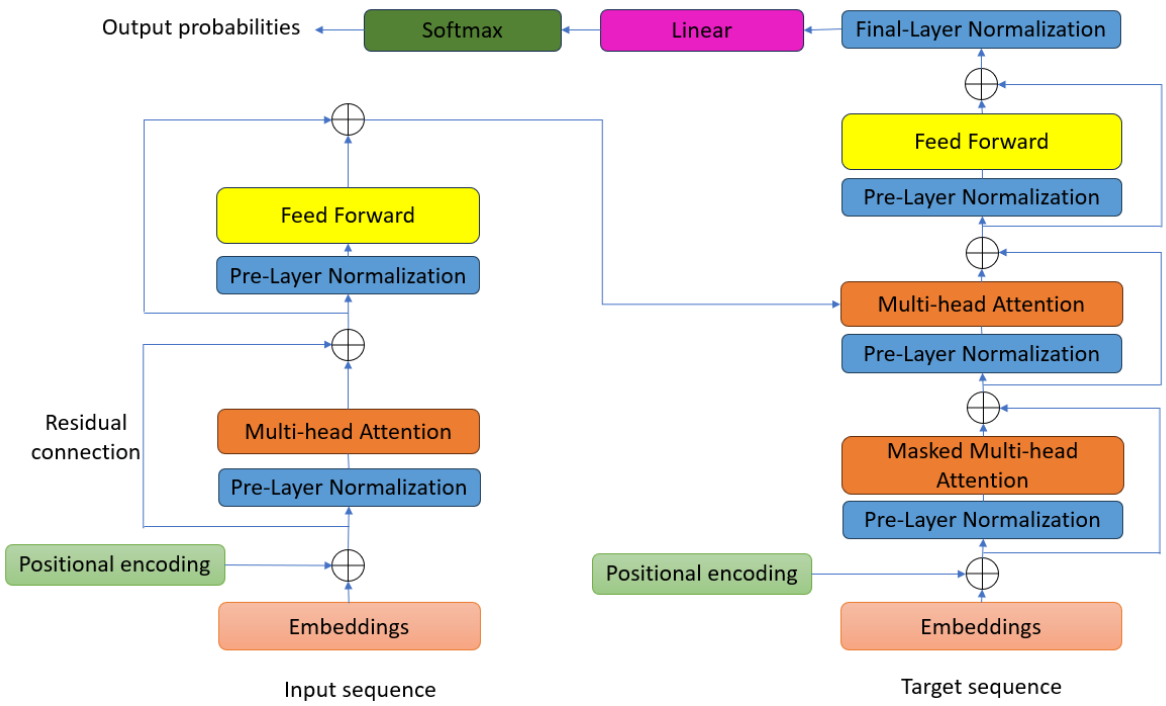
Kiến trúc Transformer được triển khai theo mô hình **Pre-Layer Normalization** với hai thành phần chính là Encoder và Decoder.

Encoder

- **Embedding Layer**: Chuyển đổi token indices thành dense vectors với chiều d_{model}.
- **Positional Encoding**: Triển khai mã hoá vị trí bằng các hàm lượng giác với các tần số khác nhau:

$$\begin{cases} PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \\ PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right) \end{cases} \quad (1)$$

  Trong đó pos là vị trí của token trong chuỗi, i là chỉ số chiều của vector nhúng, và d_{model} là số chiều của không gian biểu diễn.
  Thông tin vị trí được cộng trực tiếp vào embedding vectors, cho phép model nhận biết thứ tự của các token trong sequence.
- **Multi-Head Attention**: Triển khai scaled dot-product attention với nhiều heads hoạt động song song. Mỗi head học một không gian biểu diễn khác nhau, sau đó được concatenate và chiếu về d_{model}.
- **Pre-Layer Normalization**: Layer Normalization được *đặt trước* mỗi sublayer thay vì sau như Transformer gốc. Cấu trúc này cho phép gradient truyền trực tiếp qua residual connections, giúp training ổn định hơn.



Hình 1: Kiến trúc cơ bản của model

- **Feed-Forward Network**: Mạng hai lớp fully-connected với hàm kích hoạt ReLU: $\text{FFN}(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2$. Lớp FFN giúp tăng khả năng biểu diễn phi tuyến của mô hình.
- **Residual Connection**: Kết nối tắt cộng input với output của sublayer, kết hợp với dropout để regularization.

Decoder

- **Embedding Layer và Positional Encoding**: Được thiết kế và sử dụng tương tự Encoder.
- **Masked Multi-Head Self-Attention**: Tương tự Multi-Head Attention nhưng áp dụng causal mask (lower triangular matrix) để ngăn decoder nhìn thấy các token tương lai, đảm bảo tính auto-regressive trong quá trình sinh văn bản.
- **Cross-Attention**: Query được tạo từ decoder hidden states, trong khi Key và Value đến từ encoder output. Cơ chế này cho phép decoder "attend" vào toàn bộ source sequence để lấy thông tin ngữ cảnh phù hợp cho việc sinh token tiếp theo.
- **Linear Output Projection**: Lớp tuyến tính chiếu từ không gian d_{model} sang vocabulary size, tạo ra logits cho mỗi token trong từ điển. Output đi qua softmax để có phân phối xác suất.

3.3.2 Các Kỹ Thuật Tối Ưu

Ngoài các thành phần cơ bản, model tích hợp các kỹ thuật tối ưu sau để cải thiện hiệu suất:

Scaled Embedding Embedding vectors được scale với hệ số $\sqrt{d_{model}}$ trước khi cộng với positional encoding:

$$\text{ScaledEmbed}(x) = \text{Embedding}(x) \times \sqrt{d_{model}} \tag{2}$$

Kỹ thuật này cân bằng magnitude giữa embedding và positional encoding, ổn định gradient flow ngay từ đầu quá trình training.

Label Smoothing Loss Thay vì sử dụng one-hot targets, Label Smoothing phân phối một phần xác suất ϵ cho các class khác:

$$y_{smooth}(k) = \begin{cases} 1 - \epsilon & \text{nếu } k = y_{true} \\ \frac{\epsilon}{K-1} & \text{các class còn lại} \end{cases} \tag{3}$$

Với $\epsilon = 0.1$ và K là vocabulary size. Kỹ thuật này ngăn model trở nên quá tự tin vào predictions, cải thiện khả năng generalization.

Weight Tying Decoder embedding matrix và output projection layer chia sẻ cùng một ma trận

trọng số:

$$W_{embed} = W_{output}^{T} \tag{4}$$

Kỹ thuật này giảm khoảng 30% số parameters của decoder đồng thời cải thiện performance vì embedding và output projection học cùng một representation space.

3.4 Training Strategy

Chiến lược huấn luyện được thiết kế để đảm bảo sự ổn định của gradient và tăng khả năng tổng quát hóa của mô hình Transformer trên tập dữ liệu quy mô lớn.

Optimizer: Chúng tôi sử dụng thuật toán tối ưu AdamW (Adam with decoupled weight decay) để tách biệt việc cập nhật trọng số và cơ chế weight decay, giúp kiểm soát overfitting hiệu quả hơn.

Learning Rate Scheduler: Với dataset lớn và để giải quyết vấn đề bất ổn định gradient trong giai đoạn đầu của Transformer, chúng tôi áp dụng chiến lược học tập đa giai đoạn bao gồm:
Linear warmup (10K steps): Tăng dần learning rate từ 0 đến giá trị cực đại để tránh bùng nổ gradient và ổn định gradient flow.
Hold: Duy trì mức learning rate cực đại để tối đa hóa khả năng học các đặc trưng phức tạp từ tập dữ liệu lớn.
Cosine decay: Trong khoảng thời gian huấn luyện còn lại, learning rate giảm dần về 10% giá trị ban đầu, giúp mô hình tinh chỉnh ổn định và hội tụ mượt mà đến nghiệm tối ưu tốt hơn.

Regularization: Để giảm hiện tượng overfitting và cải thiện khả năng tổng quát hóa của mô hình, chúng tôi áp dụng một số kỹ thuật regularization quan trọng. Thay vì sử dụng phân phối mục tiêu dạng one-hot, hàm loss được tính toán dựa trên phân phối đã được làm mịn nhờ **Label Smoothing**:

$$q'(k|x) = (1-\epsilon)\delta_{k,y} + \frac{\epsilon}{V} \tag{5}$$

Trong đó V là kích thước từ vựng và ϵ là hệ số làm mịn. Kỹ thuật này phân bổ một phần nhỏ xác suất sang các token khác ngoài token ground truth, giúp mô hình tránh việc trở nên quá tự tin (overconfident), từ đó cải thiện độ ổn định và khả năng generalization.
Ngoài ra, kỹ thuật **Gradient Clipping** được áp dụng để khống chế độ lớn của gradient, ngăn chặn hiện tượng bùng nổ gradient (exploding gradients) thường gặp trong các model có kiến trúc sâu.

Loss Function: Cross-Entropy Loss với Label Smoothing

3.5 Inference Engine

Quá trình suy luận (Inference) được thiết kế để cân bằng giữa chất lượng bản dịch và chi phí tính toán thông qua hai phương pháp chính:

Greedy Decoding: Đây là phương pháp giải mã nhanh nhất, tại mỗi bước thời gian t, mô hình chọn token y_t có xác suất hậu nghiệm cao nhất:

$$y_t = \arg\max_{w \in V} P(y_t = w | y_{<t}, X) \tag{6}$$

Mặc dù hiệu quả về mặt tốc độ, phương pháp này dễ rơi vào các cực tiểu địa phương do tính chất tham lam của nó.

Beam Search with Length Penalty: Để khắc phục nhược điểm của Greedy Decoding, chúng tôi triển khai Beam Search với độ rộng k. Tại mỗi bước, hệ thống duy trì k chuỗi có xác suất tích lũy cao nhất. Cơ chế **Length Penalty** cũng được áp dụng để chuẩn hóa điểm số của các chuỗi ứng viên, nhằm tránh thiên kiến đối với các câu dịch ngắn:

$$s(Y, X) = \frac{\log P(Y|X)}{lp(Y)} \tag{7}$$

$$lp(Y) = \frac{(5 + |Y|)^{\alpha}}{(5+1)^{\alpha}} \tag{8}$$

Trong đó α là tham số điều khiển mức độ ưu tiên độ dài, giúp mô hình tạo ra các bản dịch có cấu trúc đầy đủ và tự nhiên hơn.

4 Experimental Setup

4.1 Dataset

Dự án sử dụng tập dữ liệu **PhoMT** (7), một bộ dữ liệu quy mô lớn và chất lượng cao dành cho bài toán dịch máy Anh–Việt. Nhằm đảm bảo tính khách quan và khả năng tổng quát hóa của mô hình, chúng tôi thiết kế quy trình phân tách dữ liệu và đánh giá như sau:

- **Training Set**: Sử dụng split `train` gốc. Sau quá trình tiền xử lý, tập dữ liệu còn lại khoảng 2,4 triệu cặp câu.
- **Development Set**: Split `validation` của PhoMT được chia nhỏ thành hai phần phục vụ các mục đích khác nhau: một phần dùng để theo dõi giá trị Validation Loss nhằm giám sát quá trình hội tụ; phần còn lại dùng để đánh giá điểm BLEU định kỳ sau mỗi 2 epoch. Quy trình này cho phép thực hiện *Model Selection* một cách chính xác để chọn ra checkpoint tối ưu nhất.

- **Final Evaluation Set**: Toàn bộ split test của PhoMT được giữ nguyên và cô lập hoàn toàn khỏi quá trình huấn luyện và lựa chọn model. Tập này chỉ được sử dụng một lần duy nhất để đánh giá checkpoint tốt nhất đã chọn, đảm bảo kết quả không bị nhiễu bởi hiện tượng rò rỉ dữ liệu (*data leakage*).

Data Split	Number of Samples
Train Set	$\sim$ 2,400,000
Dev set	$\sim$ 7,000
Test Set	$\sim$ 7,000
Evaluation Set	$\sim$ 15,000

Bảng 2: Thống kê số lượng mẫu dữ liệu sau tiền xử lý

4.2 Training Configuration

Các thiết lập thực nghiệm được lựa chọn nhằm tối ưu hóa hiệu suất của mô hình trên phần cứng sẵn có, đồng thời đảm bảo tính ổn định trong suốt quá trình hội tụ.

Optimization and Regularization: Chúng tôi sử dụng thuật toán AdamW với các tham số điều khiển hàm loss và quy trình regularization được thiết lập như sau:

Hyperparameter	Value
Optimizer	AdamW
β_1	0.9
β_2	0.98
ϵ	10^{-9}
Weight Decay	10^{-4}
Label Smoothing (ϵ_{ls})	0.1
Dropout Rate	0.1
Gradient Clipping	5.0

Bảng 3: Optimizer and Regularization Configuration

Training Schedule and Environment: Mô hình được huấn luyện trên quy mô lớn với chiến lược phân bổ tài nguyên và thời gian cụ thể như sau:

Setting	Value
Batch Size	48
Number of Epochs	15
Max Seq Length	256
Warmup Steps	10,000
Hardware	RTX 3090/V100
Training Time	30 hours
Precision	Full Precision (FP32)

Bảng 4: Training Configuration

4.3 Inference Configuration

Quá trình suy luận được cấu hình để tối ưu hóa sự cân bằng giữa chất lượng ngôn ngữ và tốc độ xử lý. Chúng tôi triển khai hai cơ chế giải mã chính để đánh giá hiệu năng mô hình.

Beam Search Decoding Đây là phương pháp giải mã mặc định được sử dụng để báo cáo điểm BLEU cuối cùng. Các tham số được thiết lập nhằm ưu tiên các chuỗi có xác suất tích lũy cao và độ dài hợp lý.

Parameter	Value
Beam Size	5
Max Decode Length	256
Length Penalty (α)	0.6
Early Stopping	True

Bảng 5: Beam Search Configuration

Greedy Decoding Ngoài Beam Search, cơ chế Greedy Decoding cũng được sử dụng trong các bài thử nghiệm về tốc độ suy luận (latency).

4.4 Evaluation Metrics

BLEU Score: BLEU (Bilingual Evaluation Understudy) đo độ trùng khớp n-gram giữa hypothesis và reference.

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{9}$$

Trong đó:

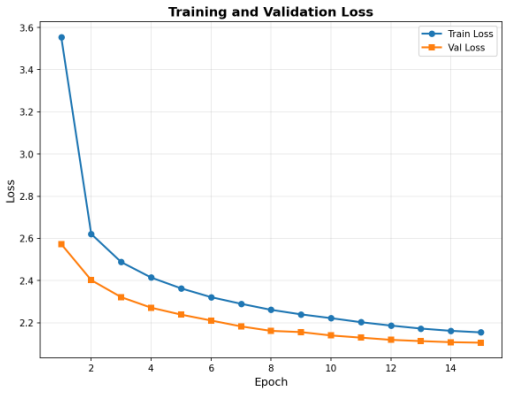
- p_n: Modified n-gram precision
- w_n: Weight cho n-gram (thường = 1/N)
- BP: Brevity Penalty để phạt câu quá ngắn

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{1-r/c} & \text{if } c \leq r \end{cases} \tag{10}$$

5 Empirical Results

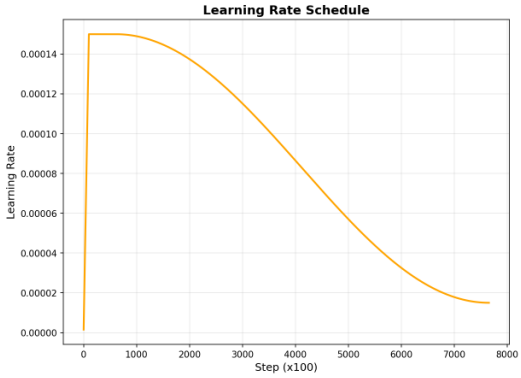
5.1 Training Progress and Checkpoint Selection

Mô hình được huấn luyện trong 15 epochs. Các kết quả huấn luyện được trình bày trong các hình dưới đây.



Hình 2: Training and validation loss across epochs

Hình 2 cho thấy training loss và validation loss đều giảm ổn định trong suốt quá trình huấn luyện. Đường validation loss bám khá sát training loss, cho thấy mô hình học tốt mà không xuất hiện hiện tượng overfitting nghiêm trọng.

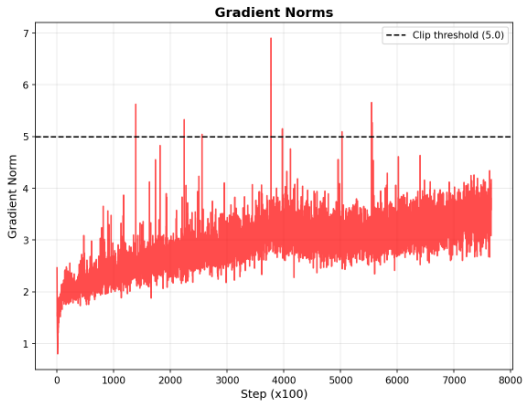


Hình 3: Learning rate schedule

Hình 3 minh họa chiến lược learning rate. Ở giai đoạn đầu, learning rate được tăng dần (warmup), giúp mô hình ổn định quá trình học ban đầu. Sau đó, learning rate được giảm dần theo schedule, hỗ trợ quá trình hội tụ mượt mà.

Hình 4 mô tả sự thay đổi của gradient norms trong quá trình huấn luyện. Gradient luôn ổn định và nằm trong ngưỡng mong muốn, chứng tỏ quá trình tối ưu không gặp hiện tượng gradient explosion và mô hình được huấn luyện một cách ổn định.

Dựa trên quy trình đánh giá định kỳ sau mỗi 2 epoch trên Internal Test set, chúng tôi theo dõi sự biến thiên của điểm BLEU để tìm ra điểm hội tụ tối ưu về mặt ngôn ngữ. Quá trình Inference là Greedy để tối ưu về mặt thời gian.



Hình 4: Gradient norms during training

Epoch	BLEU-4
1	23.21
3	25.83
5	27.02
7	27.67
9	28.35
11	28.64
13	29.08
15	29.09

Bảng 6: Checkpoints Evaluation Results

Kết quả tại Bảng 6 cho thấy tại Epoch 15, mô hình đạt điểm BLEU cao nhất trên Test set. Đây chính là checkpoint được lựa chọn cho các bước đánh giá và so sánh cuối cùng.

5.2 Performance Benchmarking

Sau khi chọn được checkpoint tối ưu, chúng tôi tiến hành đánh giá cuối cùng trên toàn bộ split test gốc của PhoMT (Evaluation Set). Đây là tập dữ liệu hoàn toàn độc lập, đóng vai trò như dữ liệu thực tế để so sánh công bằng với các hệ thống dịch thuật phổ biến hiện nay.

Metric	Greedy	Beam Search
BLEU-1	64.91	66.00
BLEU-2	53.02	53.86
BLEU-3	44.40	45.03
BLEU-4	37.81	38.27

Bảng 7: Final BLEU Score

Để đánh giá vị thế của hệ thống trong tương quan với các giải pháp hiện nay, chúng tôi tiến hành đối chiếu kết quả của hệ thống (phiên bản sử dụng Beam Search) với các Baseline uy tín. Việc so sánh này không chỉ dựa trên điểm số BLEU mà còn xem xét quy mô tham số để đánh giá tính hiệu quả của kiến trúc.

Model	Parameters	BLEU
Google Translate	-	39.86
Bing Translator	-	40.37
VinAI Translate	610M	44.29
Ours	**77M**	**38.27**

Bảng 8: Comparision Results

Dựa trên kết quả tại Bảng 8, hệ thống của chúng tôi cho thấy những tín hiệu rất khả quan khi so sánh trực tiếp với các giải pháp hàng đầu hiện nay. Điểm BLEU-4 đạt được là 38.27, một con số tiệm cận sát với các công cụ thương mại như Google Translate (39.86) và Bing Translator (40.37). Khoảng cách này chỉ rơi vào khoảng 1.5 đến 2 điểm BLEU, chứng minh rằng việc tự xây dựng và huấn luyện kiến trúc Transformer từ đầu với quy trình tiền xử lý dữ liệu chặt chẽ hoàn toàn có thể đạt được chất lượng dịch thuật tương đương các hệ thống toàn cầu.

Khi đối chiếu với VinAI Translate — hệ thống hiện đang giữ vị trí State-of-the-art (SOTA) với điểm BLEU-4 là 44.29 — mô hình của chúng tôi có mức chênh lệch nhất định nhưng lại thể hiện ưu thế vượt trội về tính hiệu quả của tham số. Trong khi VinAI Translate sử dụng kiến trúc mBART-large với quy mô lên tới 610 triệu tham số và được tinh chỉnh trên tập dữ liệu mở rộng 18 triệu cặp câu, mô hình của chúng tôi chỉ sử dụng 77 triệu tham số, tức là nhỏ hơn gần 8 lần.

Kết quả này cũng minh chứng cho khả năng tổng quát hóa tốt của mô hình khi được huấn luyện thuần túy trên tập dữ liệu PhoMT. Thay vì dựa vào các mô hình ngôn ngữ lớn đã được huấn luyện trước (pre-trained models), việc huấn luyện từ đầu giúp mô hình tập trung tối đa vào các đặc trưng ngữ pháp và từ vựng đặc thù của cặp ngôn ngữ Việt–Anh. Sự ổn định về điểm số trên tập Evaluation Set độc lập cho thấy hệ thống đã vượt qua được thách thức về sự khác biệt cấu trúc giữa một ngôn ngữ đơn lập như tiếng Việt và tiếng Anh, tạo tiền đề vững chắc cho việc triển khai các ứng dụng dịch máy hiệu suất cao với chi phí vận hành thấp.r

6 Conclusion

Trong dự án này, chúng tôi đã xây dựng thành công hệ thống dịch máy Anh-Việt hoàn chỉnh từ đầu sử dụng kiến trúc Transformer. Các đóng góp chính bao gồm:

1. **SentencePiece Tokenizer**: Cài đặt thành công BPE và Unigram Language Model từ scratch
2. **Transformer với các cải tiến**: Pre-LN, Weight Tying, Label Smoothing giúp training ổn định và hiệu quả
3. **Training Pipeline**: Hoàn chỉnh với checkpoint management, LR scheduling
4. **Kết quả**: Đạt BLEU-4 32% trên test set, comparable với các baseline

Future Work:

- Thử nghiệm với mô hình lớn hơn (Large configuration)
- Áp dụng Back-translation để augment data
- Integrate với pre-trained models (mBERT, XLM-R)
- Deploy như web service

Acknowledgments

Chúng tôi xin cảm ơn:

- Đại học Bách khoa Hà Nội (HUST) cho tập dữ liệu PhoMT
- Google Research cho bài báo Transformer
- Cộng đồng PyTorch và HuggingFace

References

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). *Attention is all you need*. [cite: 306, 307] Advances in neural information processing systems, 30.

[2] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). *Sequence to sequence learning with neural networks*. [cite: 310, 316] Advances in neural information processing systems, 27.

[3] Bahdanau, D., Cho, K., & Bengio, Y. (2014). *Neural machine translation by jointly learning to align and translate*. [cite: 317, 318] arXiv preprint arXiv:1409.0473.

[4] Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., & Liu, T. (2020). *On layer normalization in the transformer architecture*. [cite: 319, 320] International Conference on Machine Learning.

[5] Press, O., & Wolf, L. (2017). *Using the output embedding to improve language models*. [cite: 321] arXiv preprint arXiv:1608.05859.

[6] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). *Rethinking the inception architecture for computer vision*. [cite: 322, 323] IEEE conference on computer vision and pattern recognition.

[7] Doan, T. L., et al. (2021). *PhoMT: A High-Quality and Large-Scale Benchmark Dataset for Vietnamese-English Machine Translation*. [cite: 324, 325] arXiv preprint.

[8] Sennrich, R., Haddow, B., & Birch, A. (2016). *Neural machine translation of rare words with subword units*. [cite: 326, 327] arXiv preprint arXiv:1508.07909.

[9] Kudo, T., & Richardson, J. (2018). *SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing*. [cite: 328, 329] arXiv preprint arXiv:1808.06226.

[10] Loshchilov, I., & Hutter, F. (2017). *Decoupled weight decay regularization*. arXiv preprint arXiv:1711.05101. (Sử dụng cho thuật toán AdamW)

[11] Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., & Zettlemoyer, L. (2020). *Multilingual denoising pre-training for neural machine translation*. [cite: 292] Transactions of the Association for Computational Linguistics, 8, 726-742. (Kiến trúc mBART của VinAI Translate)

[12] Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002). *Bleu: a method for automatic evaluation of machine translation*. Proceedings of the 40th annual meeting of the Association for Computational Linguistics.