

HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.



**ĐẠI HỌC
BÁCH KHOA HÀ NỘI**
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

Attention và Transformers

ONE LOVE. ONE FUTURE.

Nội dung chính

- Seq2Seq Problems
 - Word Embedding
 - Encoder-Decoder Architecture
- Attention Mechanisms
 - Queries, Keys and Values
 - Bahdanau Attention
 - Dot Product Attention
- Transformer
 - Multi-Head Attention, Positional Encoding, Position-wise FFN
 - BERT, GPT
 - Vision Transformer
- Ứng dụng bài toán thực tế



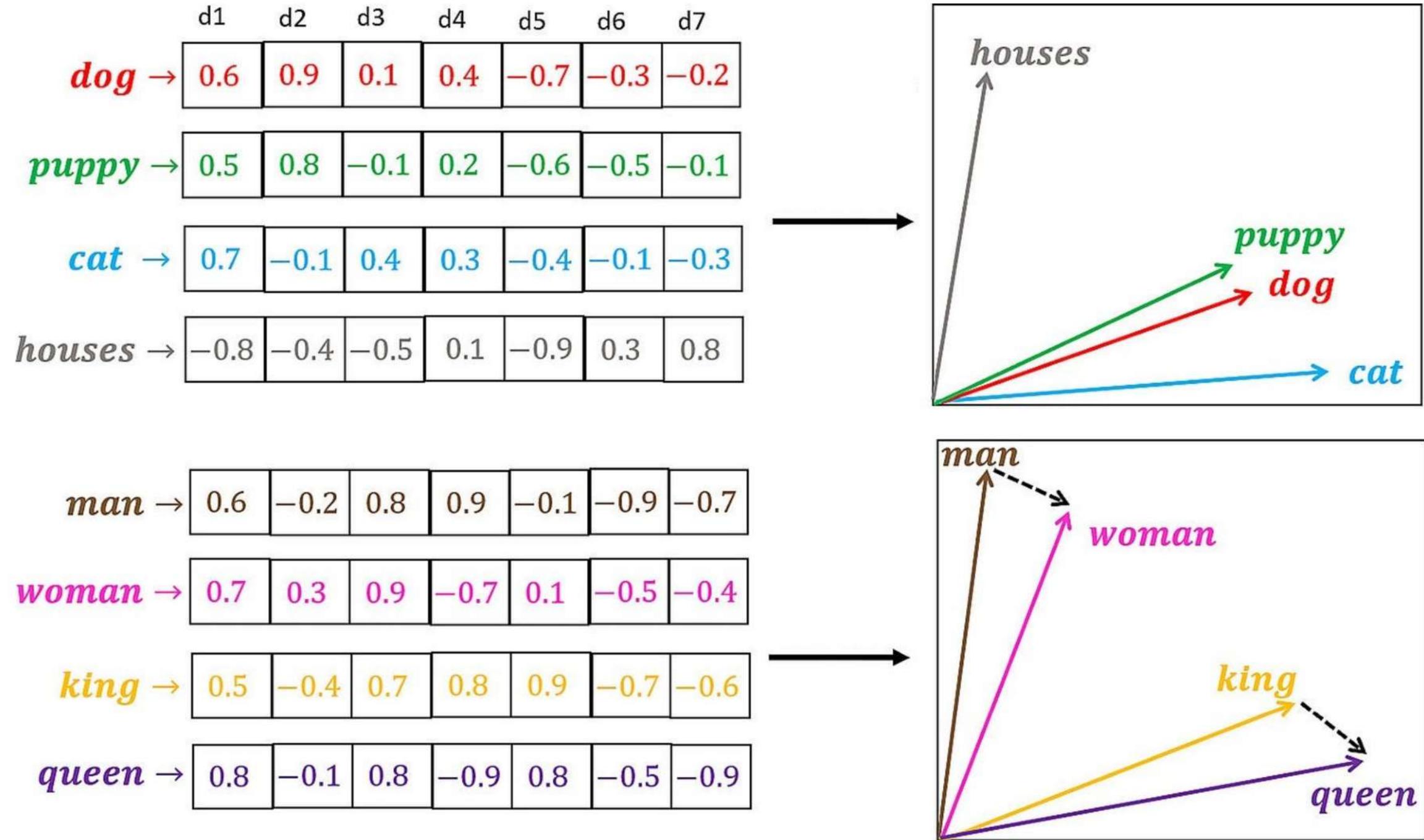
Seq2Seq Problems

- Một trong những ứng dụng lớn của RNN thời kỳ đầu chính là dịch máy, khi mô hình nhận vào một câu ở ngôn ngữ A và phải dự đoán câu tương ứng ở ngôn ngữ B
- Các câu ở hai ngôn ngữ có thể khác độ dài, và các từ tương ứng trong hai câu không nhất thiết xuất hiện theo cùng thứ tự, do sự khác biệt về cấu trúc ngữ pháp giữa hai ngôn ngữ
- Nhiều bài toán khác tương tự với mục tiêu tạo một ánh xạ từ câu này sang câu khác, được gọi chung là Seq2Seq Problems



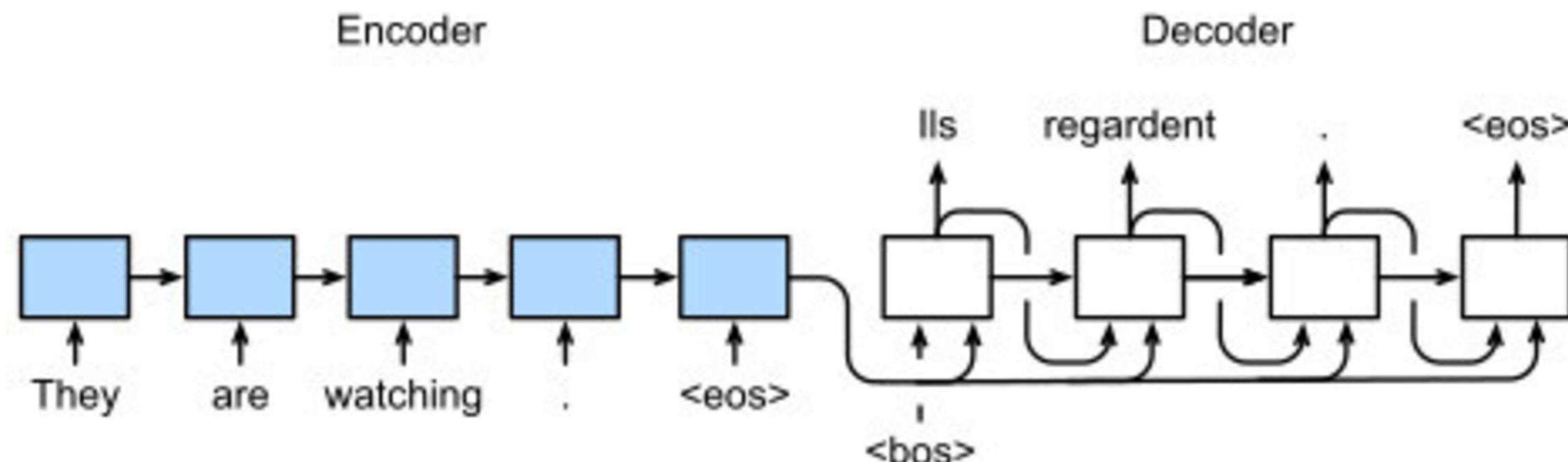
Word Embedding

- Biểu diễn một từ bằng một vector số thực trong không gian nhiều chiều, giữ lại thông tin của từ:
 - Nghĩa giống nhau thì gần nhau
 - Nghĩa khác nhau thì xa nhau



Encoder-Decoder Architecture

- Trong các bài toán Seq2Seq, chuỗi đầu vào và đầu ra thường có độ dài khác nhau, khi đó một cách tiếp cận phổ biến là sử dụng kiến trúc Encoder-Decoder

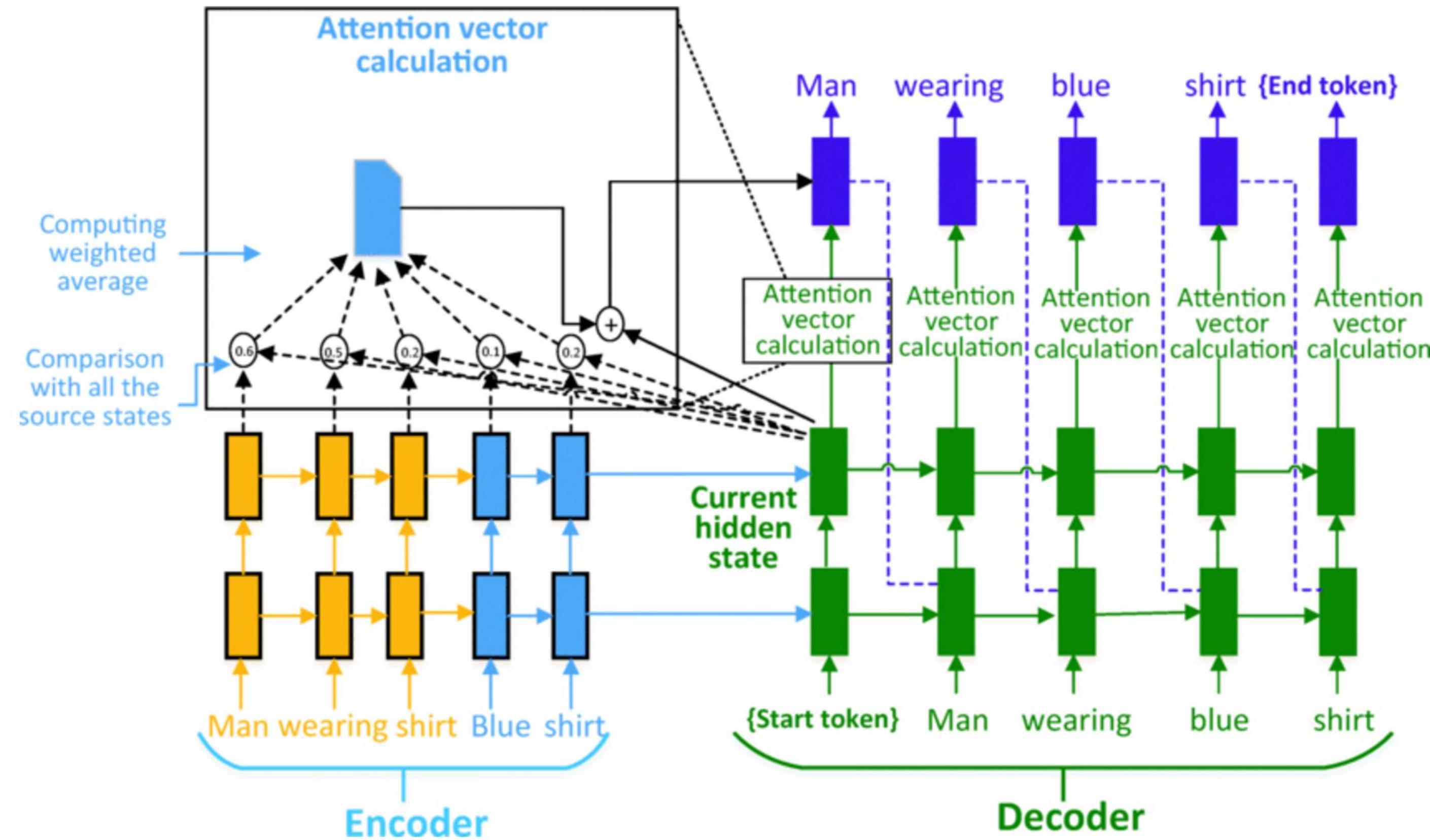


Encoder-Decoder Architecture

- Hạn chế của kiến trúc Encoder-Decoder:
 - Tắc nghẽn ngữ cảnh: Encoder nén toàn bộ thông tin câu và một context vector rồi mới truyền cho Decoder
 - Phụ thuộc tuần tự: Encoder và Decoder xử lý lần lượt từng time steps, huấn luyện chậm, không song song được

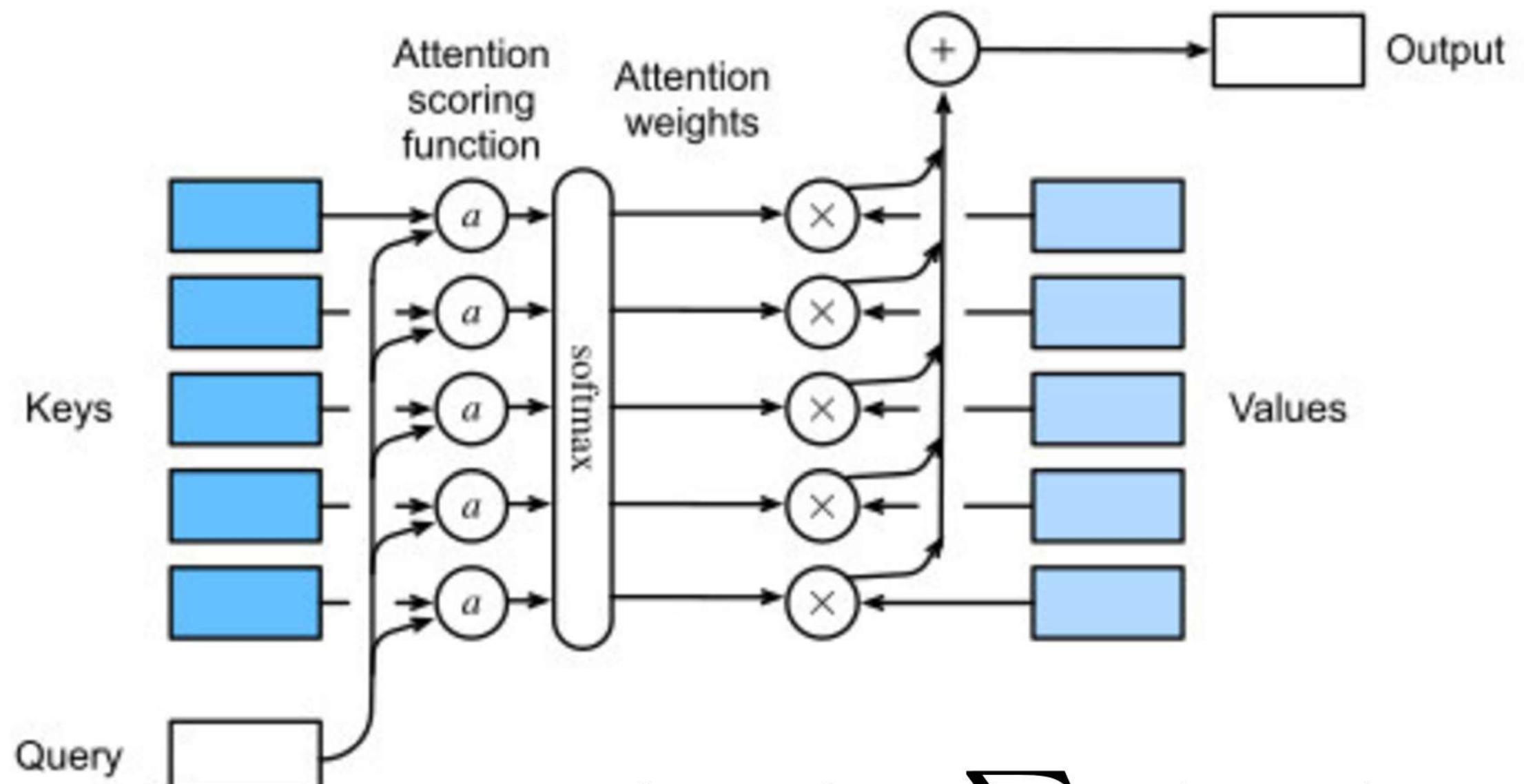


Attention Mechanisms



Queries, Keys and Values

- Attention sử dụng Query (Q), Key (K) và Value (V) lấy cảm hứng từ hệ thống truy xuất thông tin trong đó:
 - Q là câu hỏi cần tìm
 - K là các đặc trưng để so sánh
 - V là nội dung được truy xuất



$$score(Q, K) = \sum \alpha(Q, K)V$$

Bahdanau Attention

- Công thức tính Bahdanau Attention Score:

$$c_i = \sum_{j=1}^T \alpha(s_{i-1}, h_j) h_j \quad e_{ij} = v^T \tanh(W_s s_{i-1} + W_h h_j)$$
$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_k \exp(e_{ik})}$$



Dot Product Attention

$$c_i = \sum_{j=1}^T \alpha(s_{i-1}, h_j) h_j \quad \alpha_{ij} = \frac{\exp(e_{ij})}{\sum_k \exp(e_{ik})}$$

- Công thức tính Gaussian Attention Score:

$$e_{ij} = -\frac{1}{2} \|s_{i-1} - h_j\|^2 = s_{i-1}^T h_j - \frac{1}{2} \|s_{i-1}\|^2 - \frac{1}{2} \|h_j\|^2$$

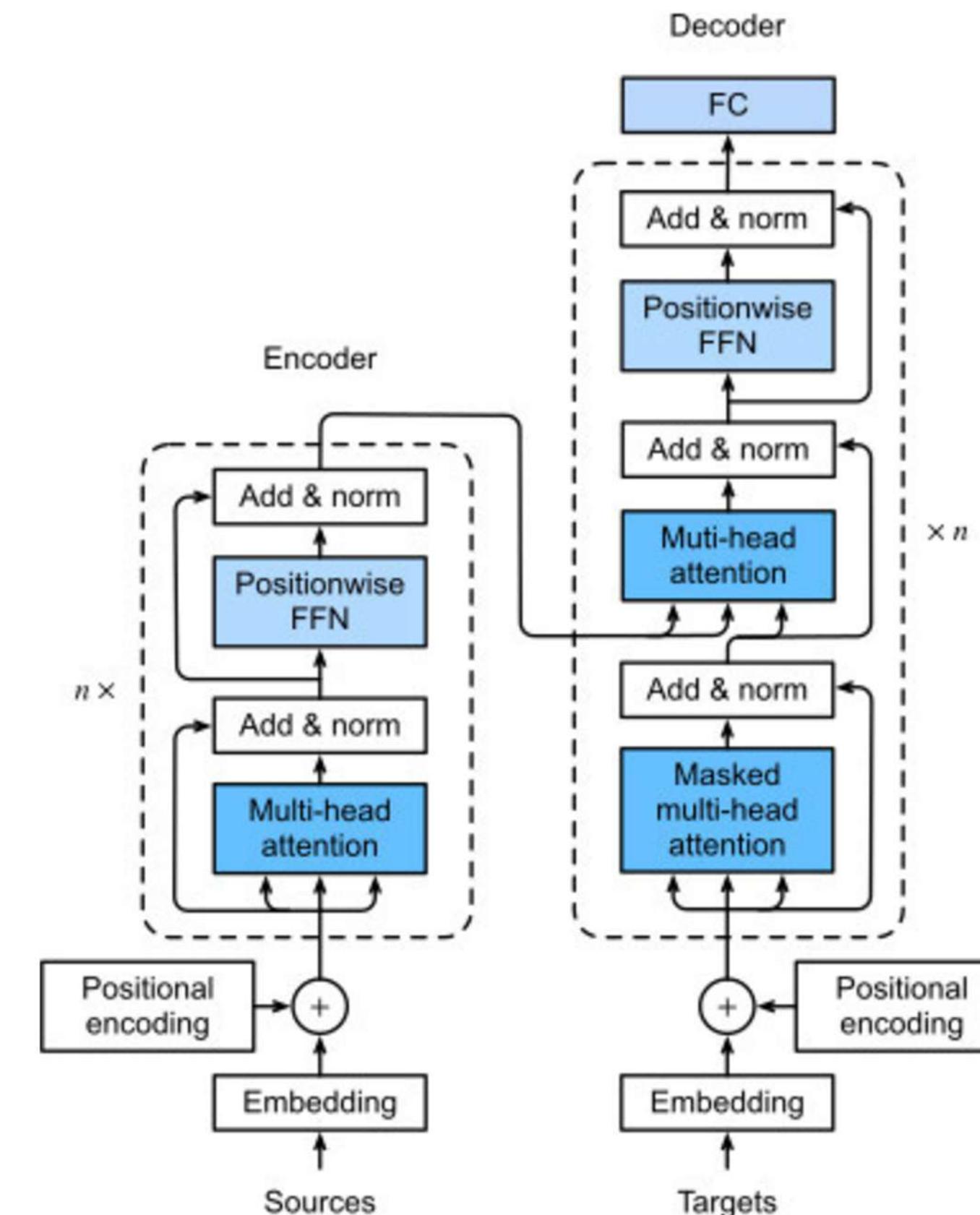
- Công thức tính Dot Product Attention Score:

$$e_{ij} = s_{i-1}^T W h_j$$

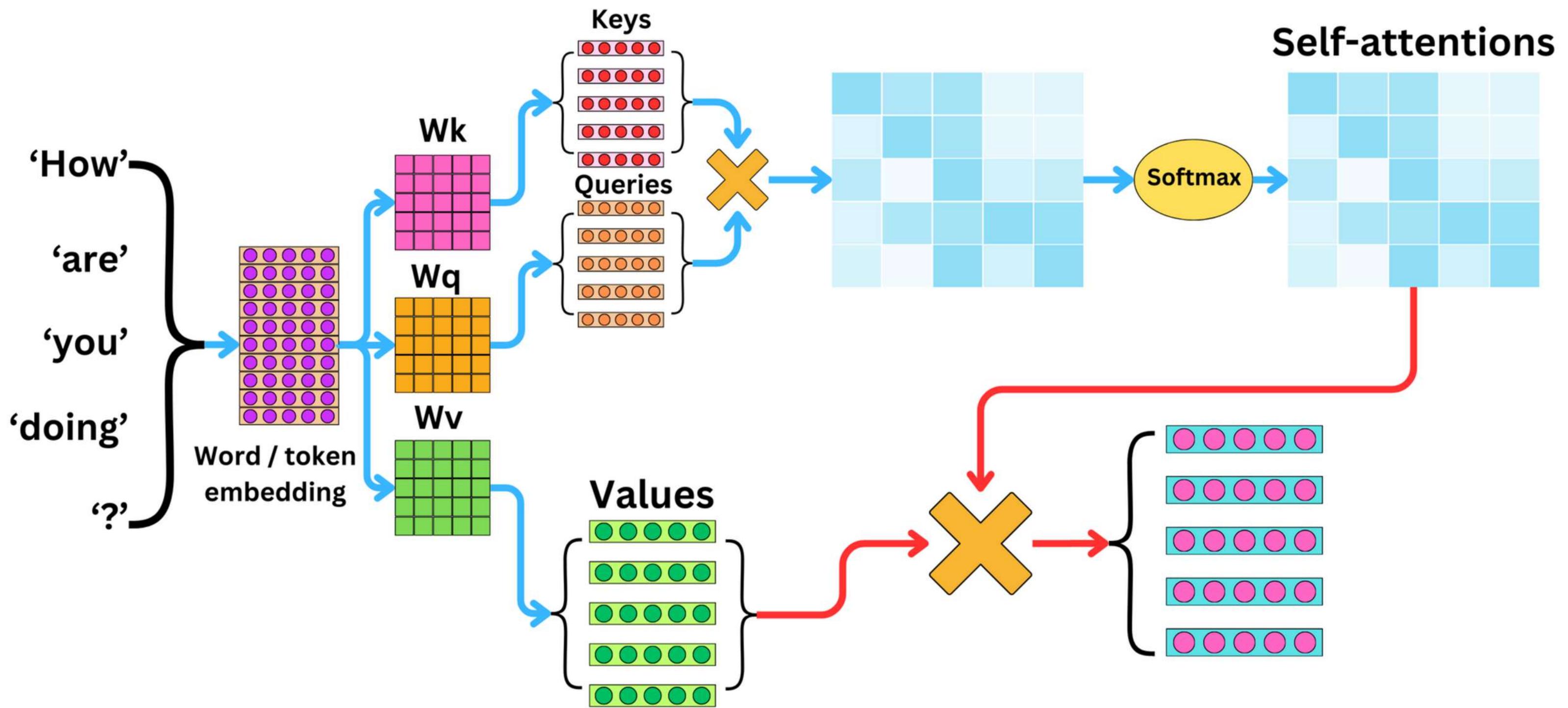


Transformer

- Transformer ra đời năm 2017, bỏ toàn bộ RNN, chỉ sử dụng Attention kết hợp với Positional Encoding



Self-Attention



Scaled Dot Product Attention

- Công thức tính Dot Product Attention Score:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{Q^T K}{\sqrt{d_k}}\right)V$$

- Trong đó: $X \in R^{d_x \times d_y}$

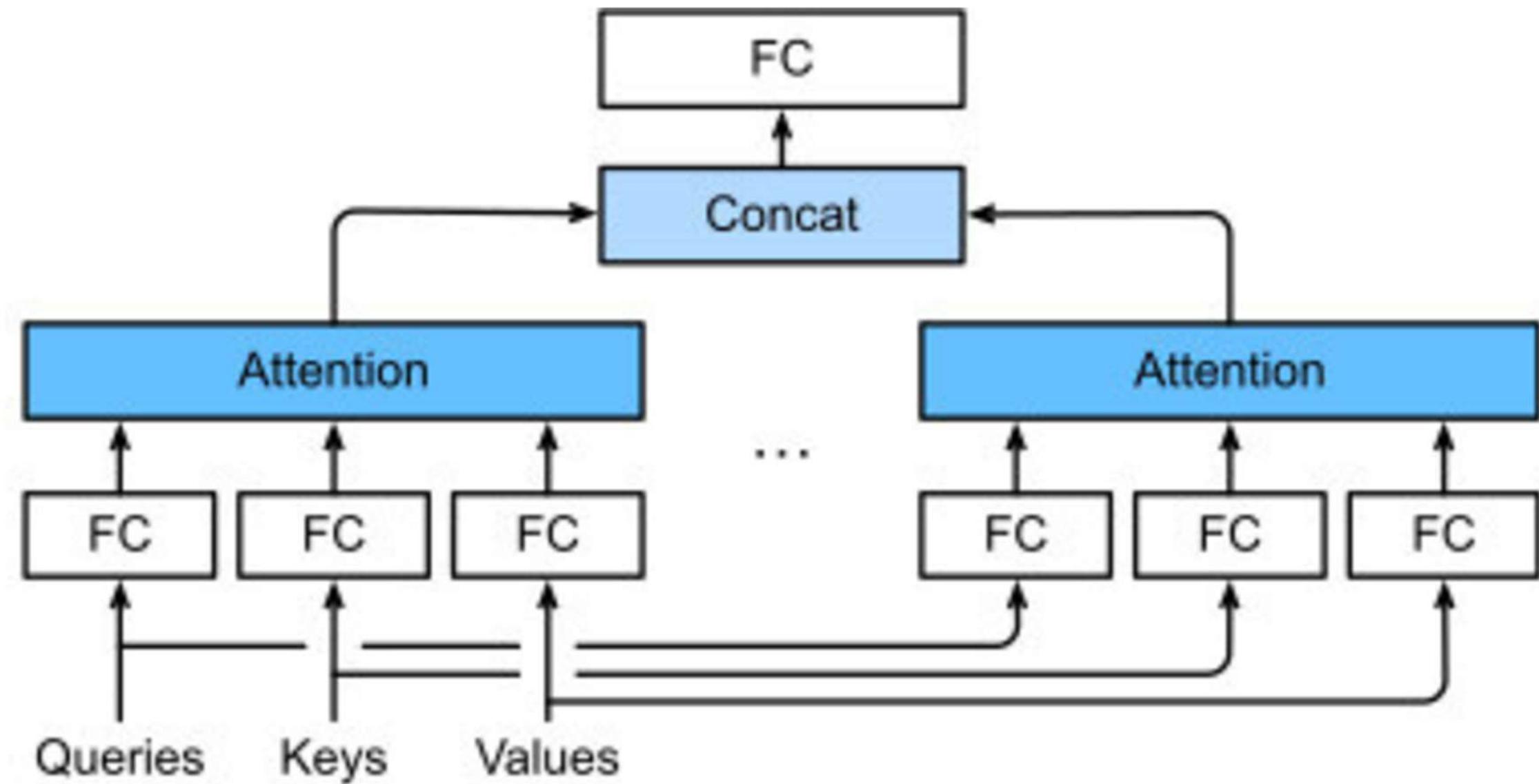
$$Q = XW_Q \in R^{d_x \times d_k}$$

$$K = XW_K \in R^{d_x \times d_k}$$

$$V = XW_V \in R^{d_x \times d_v}$$



Multi-Head Attention



$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \dots, \text{head}_h)W_O \\ \text{head}_i &= \text{Attention}(XW_Q^i, XW_K^i, XW_V^i) \end{aligned}$$

Positional Encoding

- Positional Encoding (PE) là một vector bổ sung được thêm vào embedding của mỗi token để mã hóa thông tin vị trí token trong chuỗi

$$PE_{pos,2i} = \sin\left(\frac{pos}{10000^{\frac{2i}{d}}}\right)$$

$$PE_{pos,2i+1} = \cos\left(\frac{pos}{10000^{\frac{2i}{d}}}\right)$$

- Ma trận PE không bắt buộc phải có công thức tính cố định, có thể coi PE như một ma trận embedding và được cập nhật thông qua backprop



Position-wise FFN

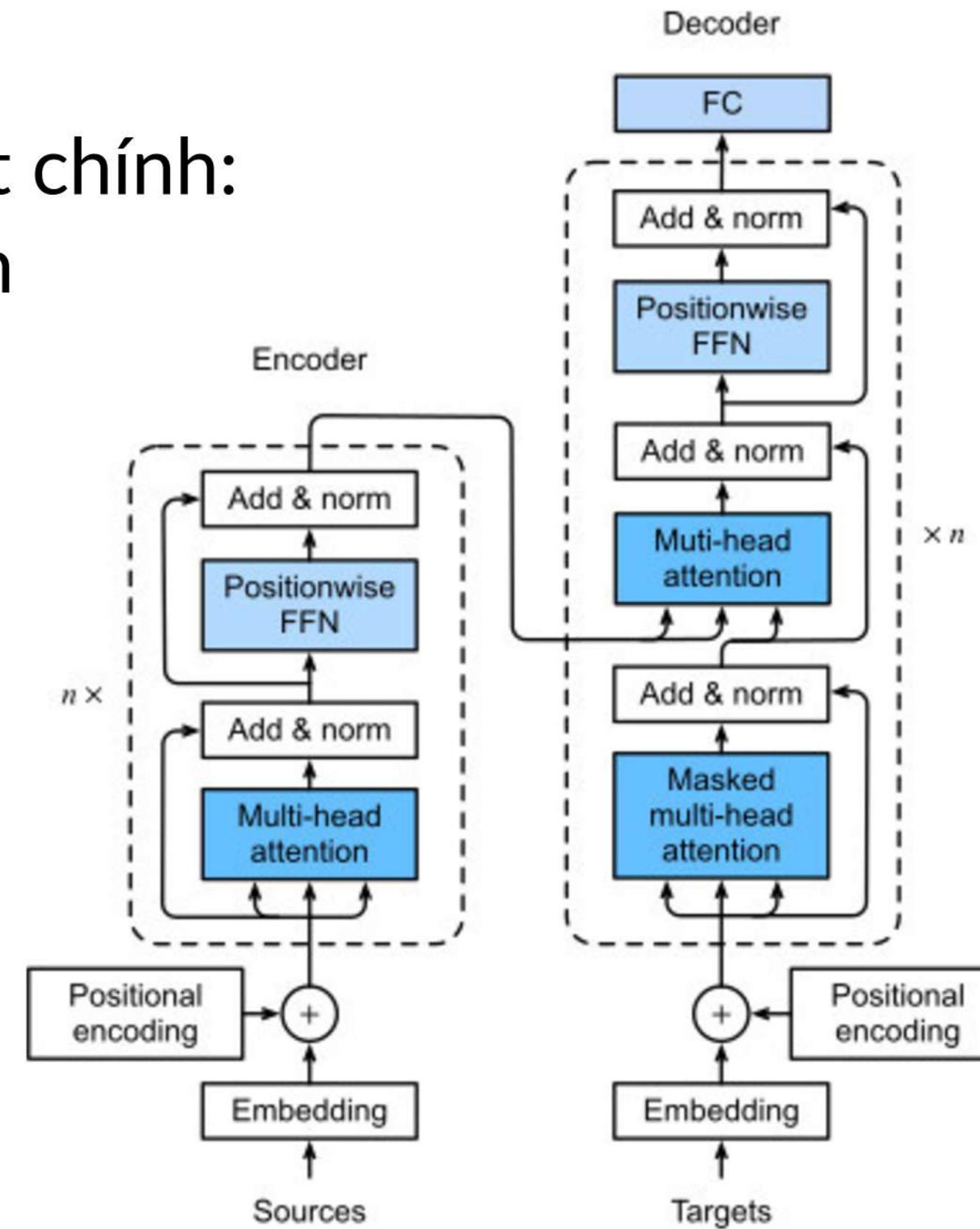
- Position-wise Feed-Forward Network áp dụng cùng một MLP nhỏ cho mỗi hàng của ma trận đầu vào

$$FFN(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2$$

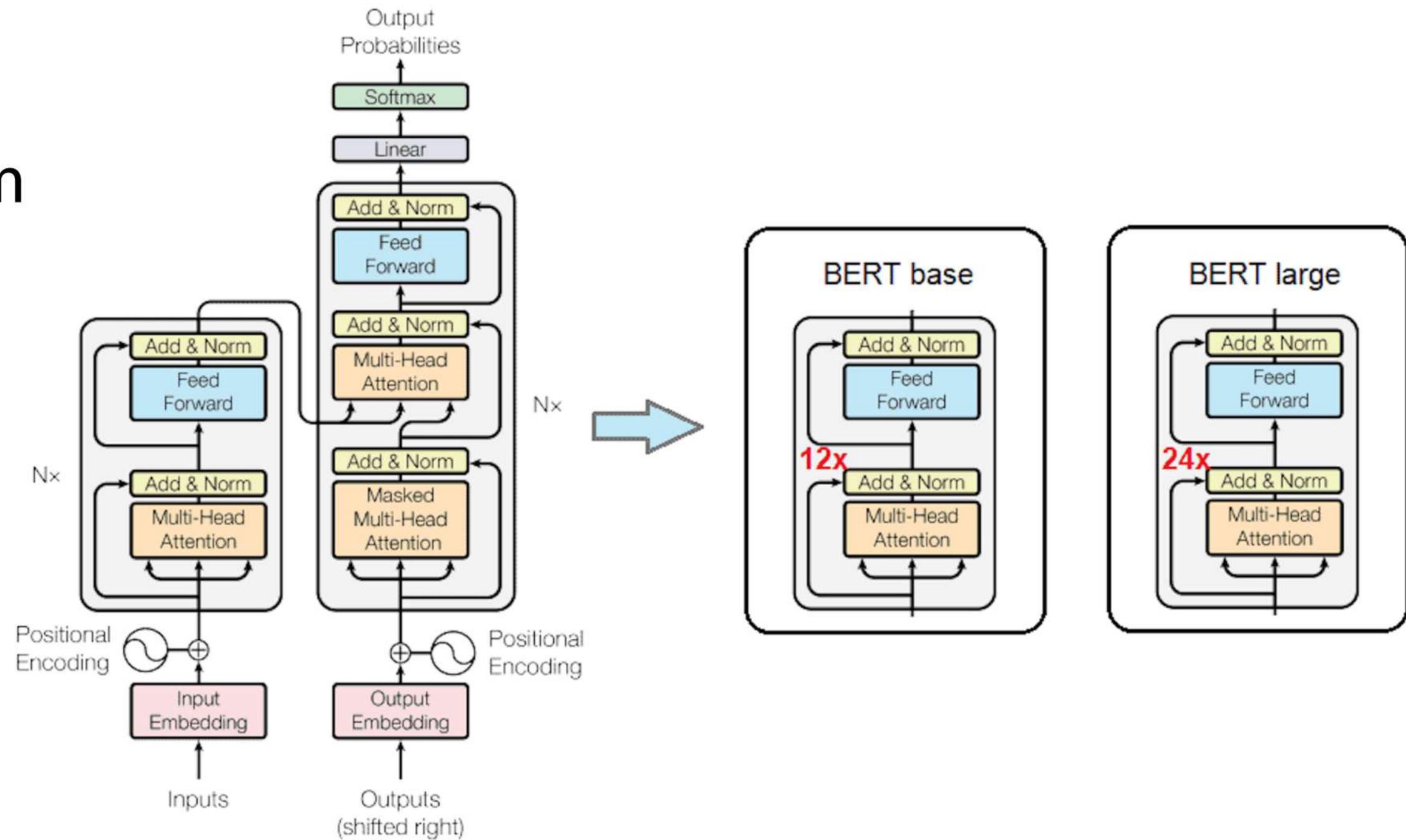
$$X = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_d \end{bmatrix}, FFN(X) = \begin{bmatrix} FFN(x_1) \\ FFN(x_2) \\ \dots \\ FFN(x_d) \end{bmatrix}$$

- FFN phi tuyến hóa embedding của từng token riêng lẻ sau khi mỗi token đã chứa thông tin tổng hợp từ toàn bộ sequence nhờ attention

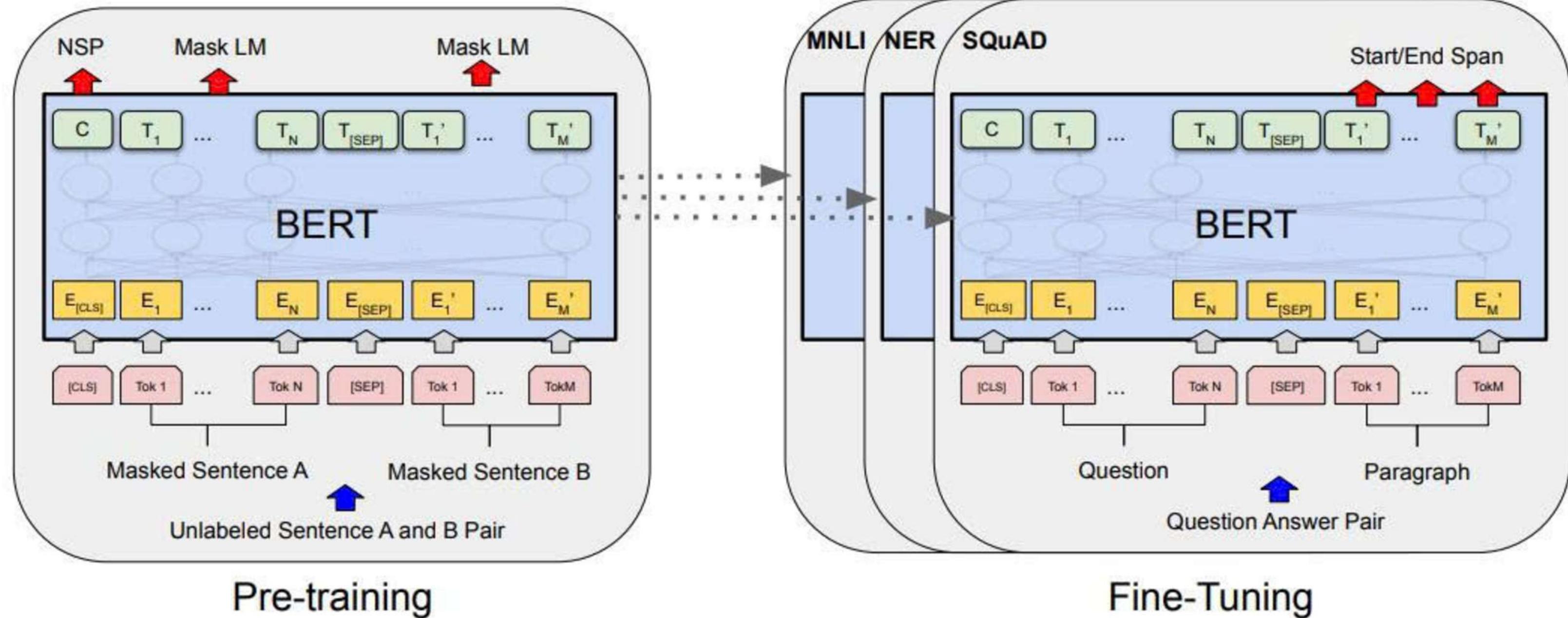
- Phần Decoder có hai sự khác biệt chính:
 - Masked Multi-head Attention
 - Cross Attention



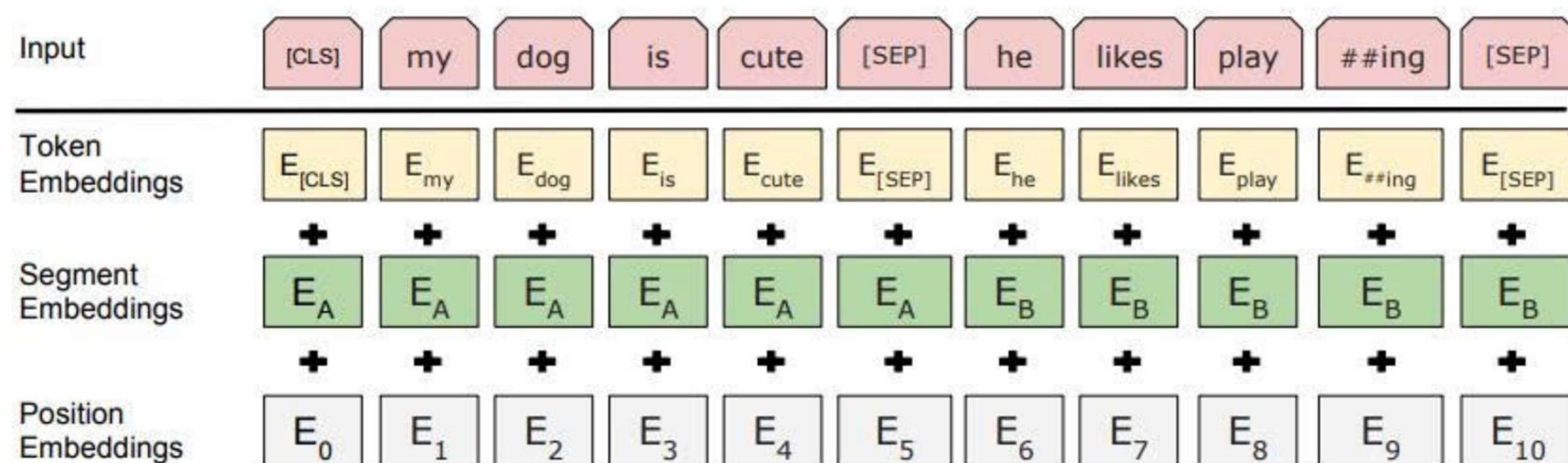
- BERT (Bidirectional Encoder Representations from Transformers) tạo ra một mô hình ngôn ngữ tổng quát, hiểu ngữ nghĩa và ngữ pháp hai chiều của câu, để tái sử dụng cho mọi tác vụ NLP khác



BERT



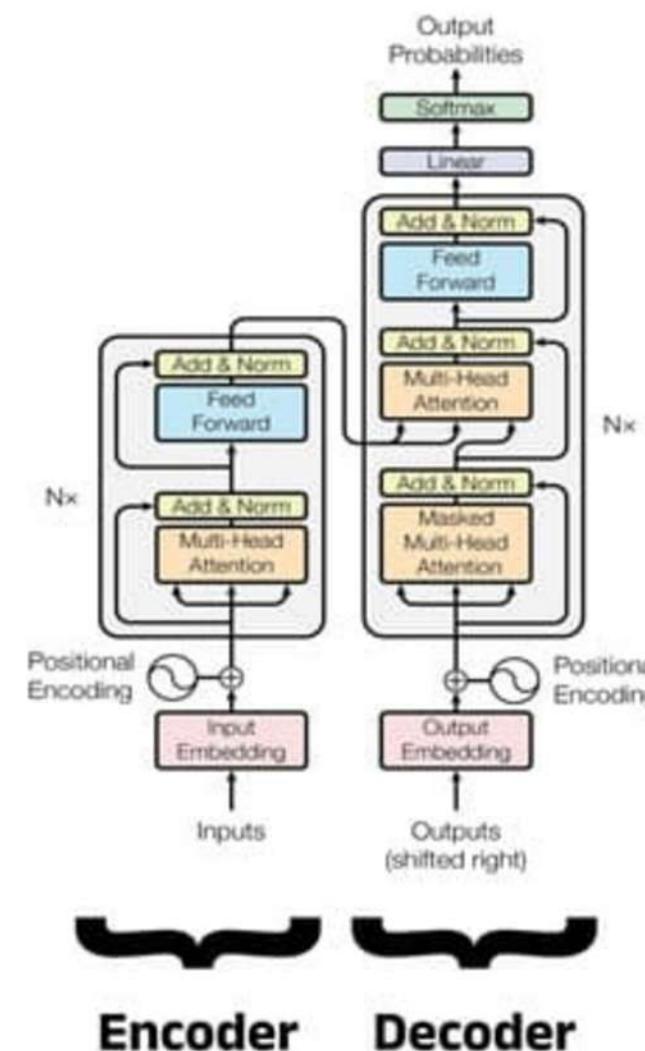
- Đầu vào của BERT gồm 2 đặc điểm chính:
 - Token [CLS] đứng đầu phục vụ cho bài toán phân loại văn bản
 - Cặp câu được gộp vào làm một và ngăn cách bởi token [SEP]



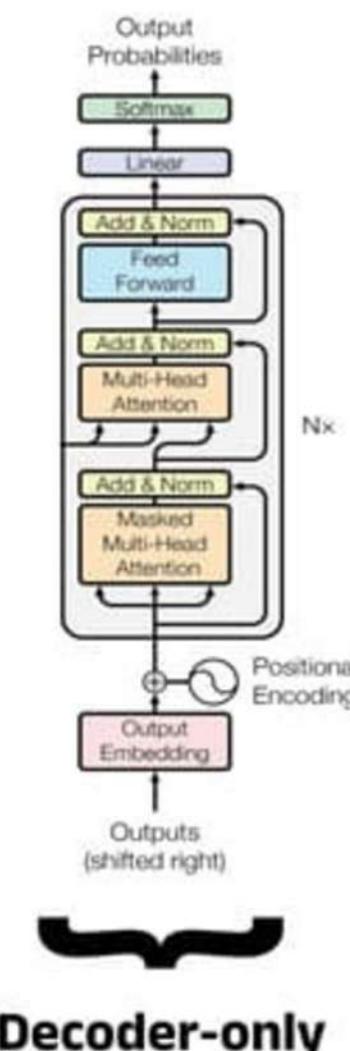
- BERT thực hiện pre-training trên hai task với mục đích học hiểu ngữ nghĩa hai chiều và hiểu quan hệ giữa các câu:
 - Masked Language Modeling (MLM)
 - Next Sentence Prediction (NSP)
- MLM: Che ngẫu nhiên 15% token trong chuỗi đầu vào bằng token [MASK], để BERT học cách dự đoán từ bị che dựa vào ngữ cảnh hai chiều
- NSP: Ghép hai câu A và B với nhau và BERT học hiểu quan hệ giữa hai câu và dự đoán B có là phần tiếp theo của A

- GPT (Generative Pre-trained Transformer) là một mô hình ngôn ngữ được thiết kế để tạo ra văn bản

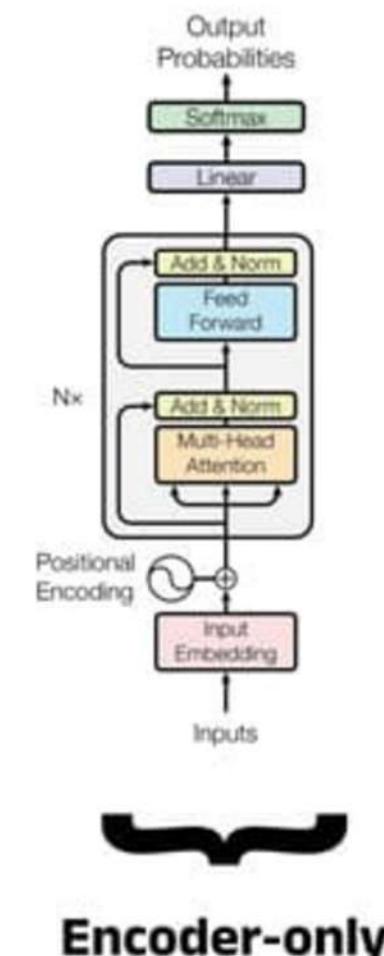
Transformer



GPT*

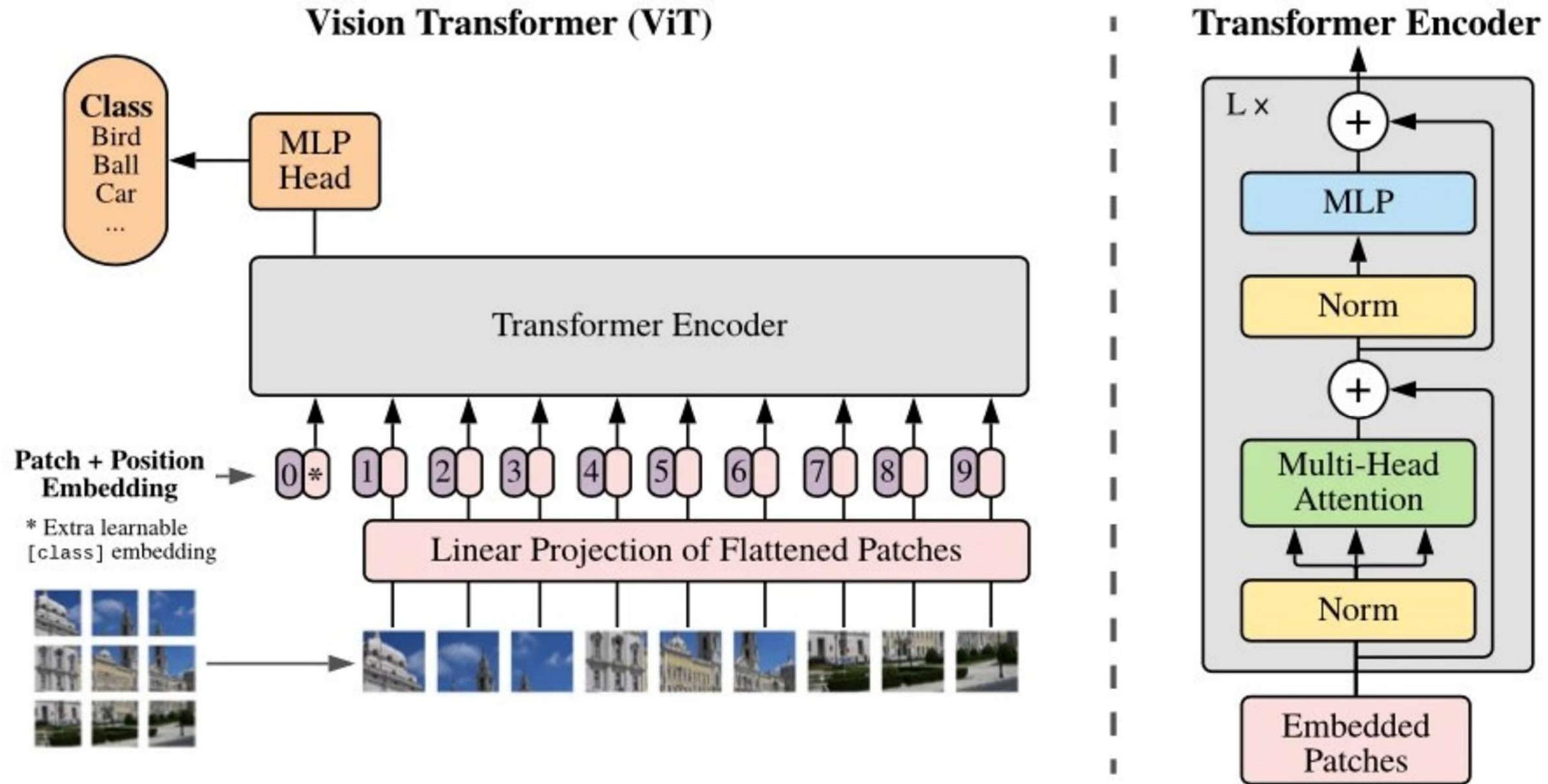


BERT*



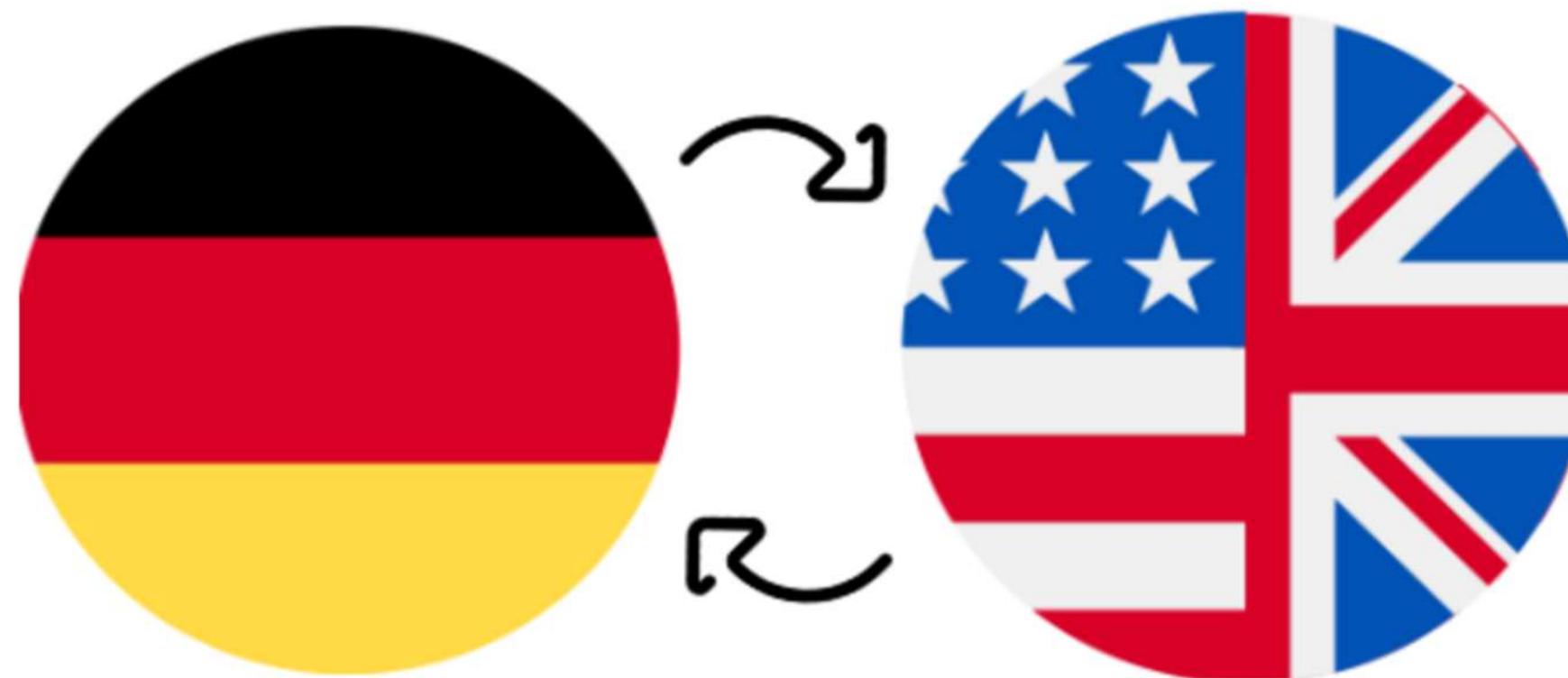
- GPT thực hiện pre-training trên một task là Next Token Prediction (NTP) với mục đích học phân phối xác suất của từ kế tiếp trong chuỗi ngôn ngữ tự nhiên
- NTP: GPT nhận toàn bộ chuỗi và dự đoán toàn bộ token kế tiếp cùng lúc, hàm mất mát được tính cho từng vị trí dự đoán so với token thật rồi cộng tổng lại

Vision Transformer



Ứng dụng trong bài toán thực tế

- Bộ dữ liệu: WMT14 en-de (Workshop on Machine Translation 2014)
- Dữ liệu chia làm 3 phần:
 - Train: 4.45 triệu cặp câu Anh - Đức
 - Validation: 2999 cặp câu Anh - Đức
 - Test: 3002 cặp câu Anh - Đức



Ứng dụng trong bài toán thực tế

- Thông tin huấn luyện: Tiny Transformer
 - max_len: 128
 - d_model: 256
 - encoder/decoder layers: 3/3
 - heads: 4
 - feedforward_dim: 512
 - dropout_rate: 0.1
 - optimizer: Adam
 - learn_rate: Noam Scheduler

$$lr = d_{model}^{-0.5} \min \left(iter^{-0.5}, iter \times warmup^{-0.5} \right)$$



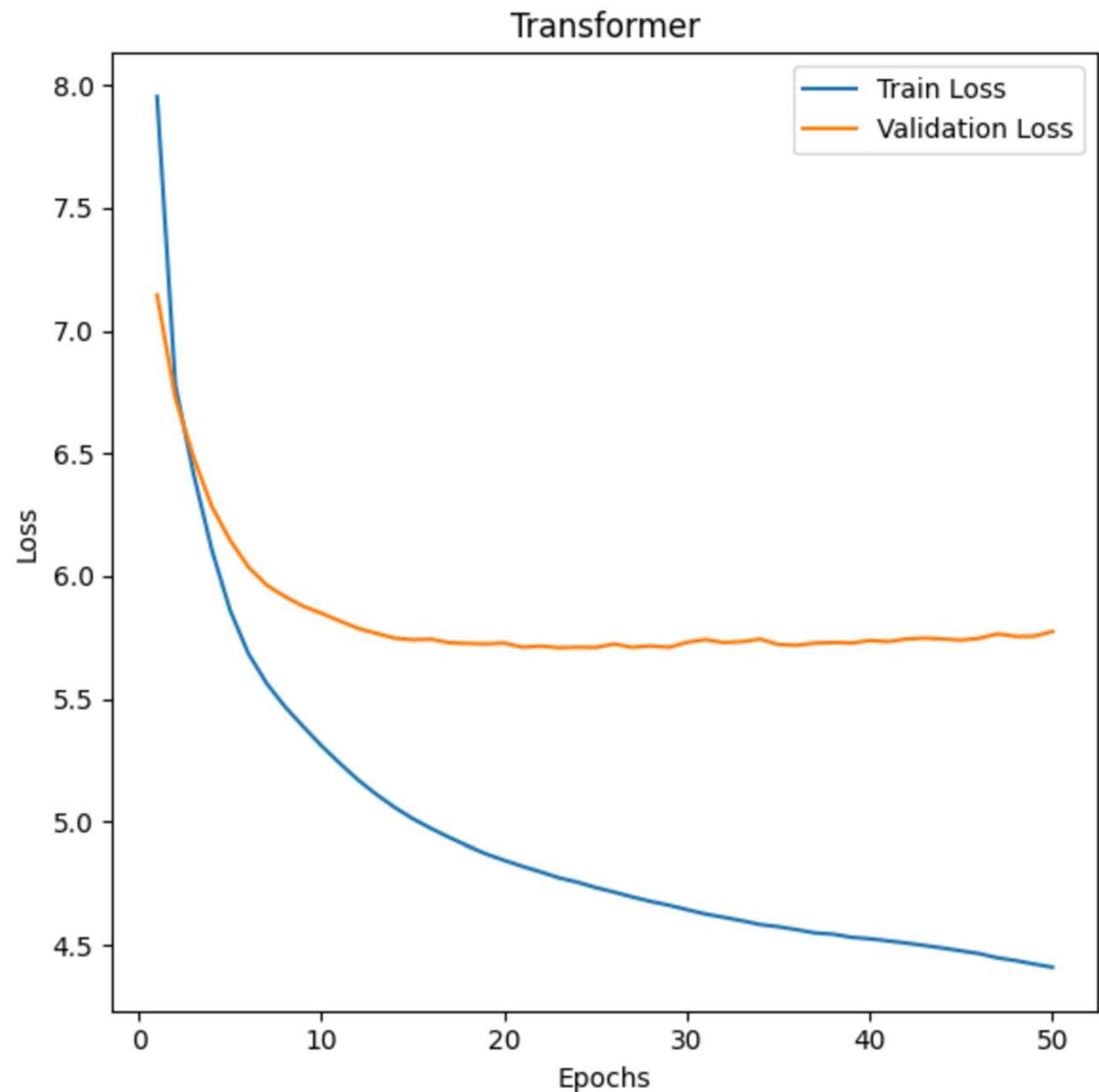
Ứng dụng trong bài toán thực tế

- BLEU (Bilingual Evaluation Understudy) là thước đo được sử dụng để đánh giá chất lượng của một mô hình dịch máy
- BLEU đo mức độ giống nhau giữa câu dịch của mô hình và một hoặc nhiều câu dịch chuẩn dựa trên 3 thành phần chính:
 - n-gram precision: mô hình dịch đúng bao nhiêu cụm từ
 - clipping: tránh "ăn gian" bằng cách lặp từ nhiều lần
 - brevity penalty: phạt nếu câu dịch ngắn hơn câu chuẩn quá nhiều



Ứng dụng trong bài toán thực tế

- Best BLEU Score: 7.42
- Best BERTScore: 0.3562

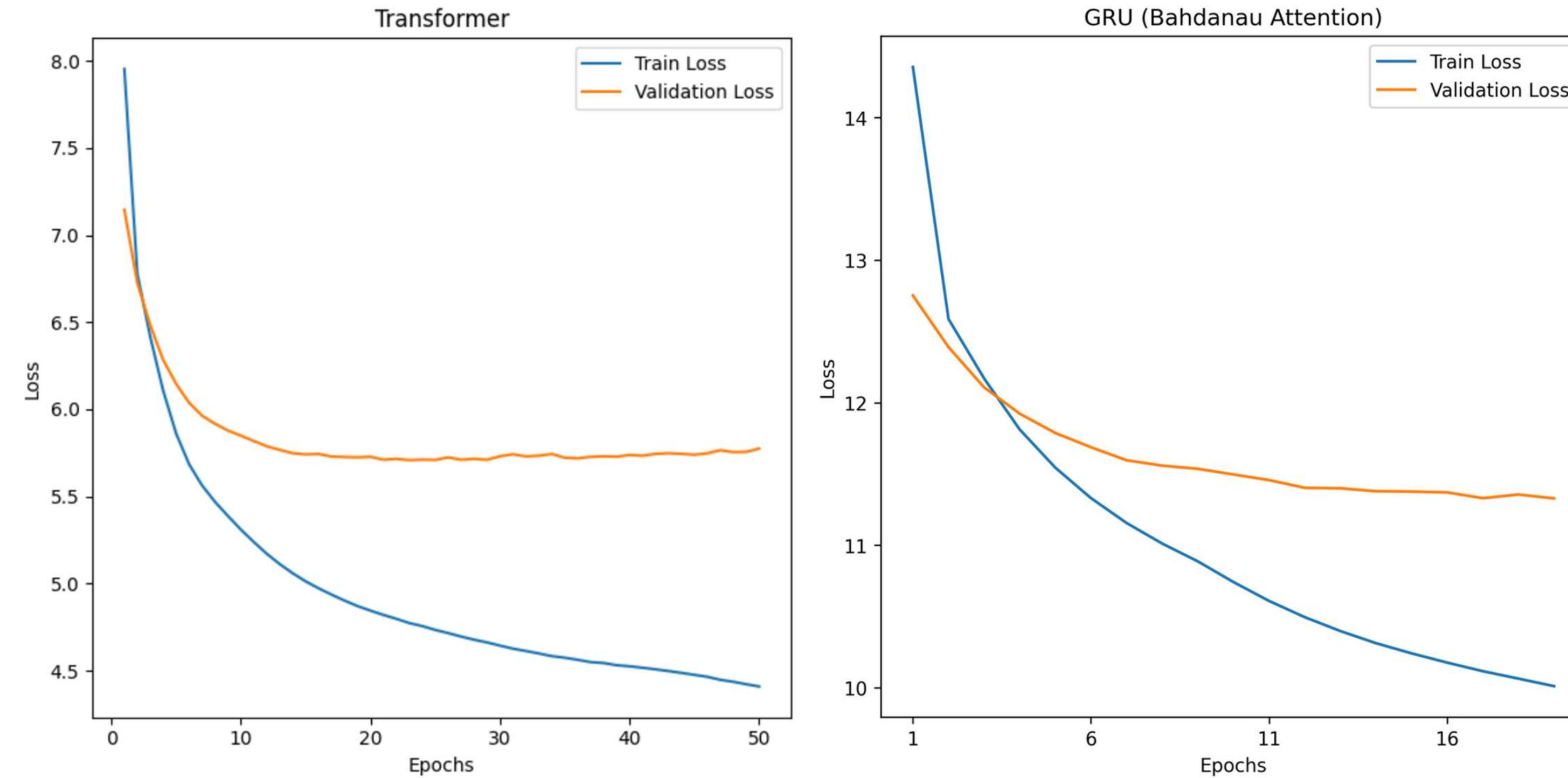


Ứng dụng trong bài toán thực tế

Model	Test Loss	BLEU Score (0-100)	BERTScore (0-1)
Tiny Transfromer	5.9083	7.42	0.3562
GRU (Bahdanau Attention)	11.3288	1.46	0.0597

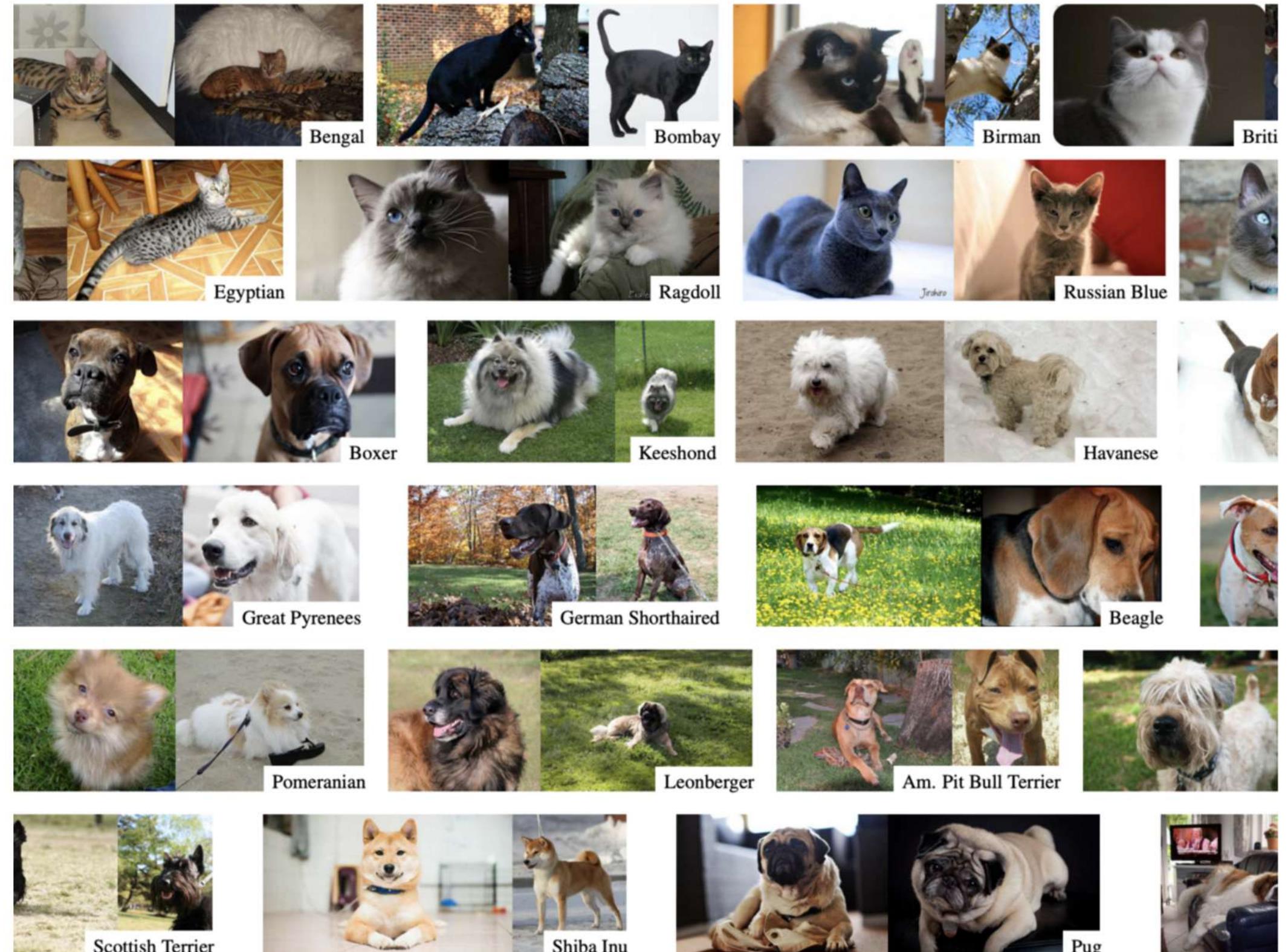


Ứng dụng trong bài toán thực tế



Ứng dụng trong bài toán thực tế

- Bộ dữ liệu:
Oxford-IIIT Pet
Dataset
- Bộ dữ liệu bao
gồm ảnh của 37
giống chó mèo
khác nhau, mỗi
loại có khoảng
200 ảnh



Ứng dụng trong bài toán thực tế

- Thông tin huấn luyện: ViT-Tiny
 - patch_size: 4
 - d_model: 32
 - n_layers: 6
 - heads: 2
 - optimizer: Adam
 - learn_rate: 0.001

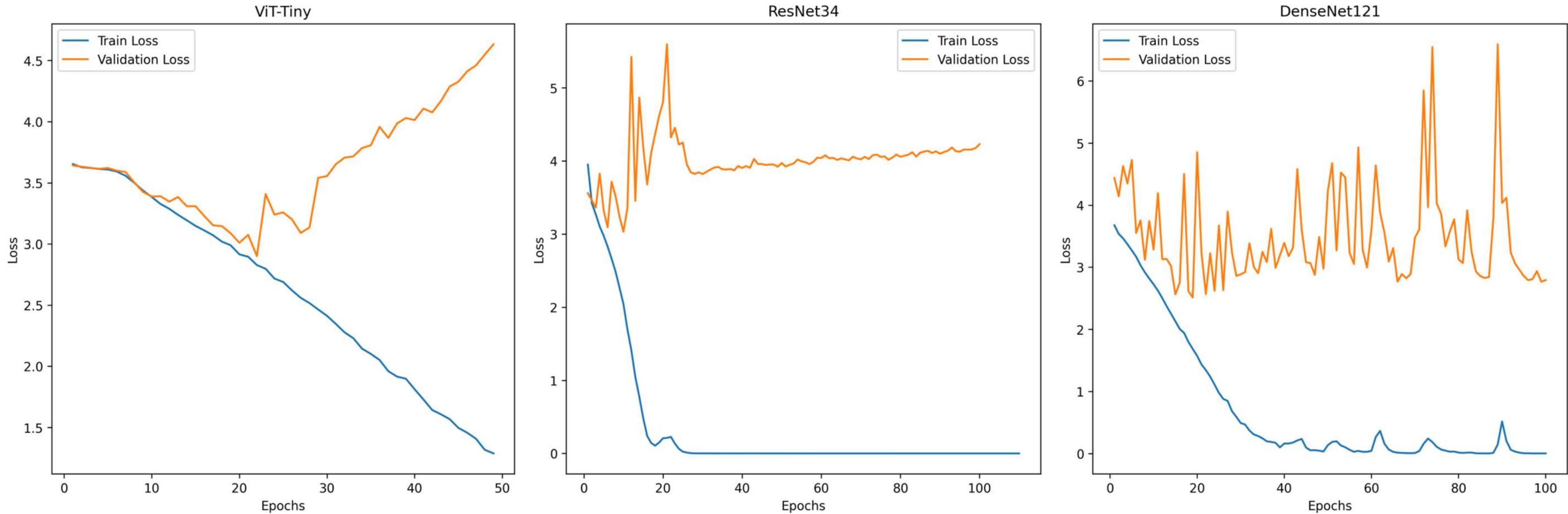


Ứng dụng trong bài toán thực tế

Model	Validation Loss	Validation Accuracy	Test Loss	Test Accuracy
ViT-Tiny	3.3082	33.21%	3.2874	36.22%
ResNet34	3.4534	31.25%	3.6776	29.88%
DenseNet121	2.5125	49.59%	3.0682	40.17%



Ứng dụng trong bài toán thực tế



A large, faint watermark of the HUST logo is visible across the entire background of the slide.

HUST

THANK YOU !