

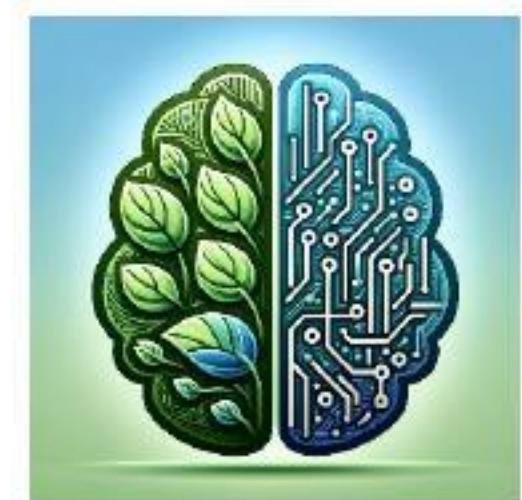


# **Natural Language Processing with Deep Learning**

## **Bài 10: MÔ HÌNH GOOGLE BERT**



[https://csc.edu.vn/data-science-machine-learning/natural-  
language-processing-with-deep-learning\\_293](https://csc.edu.vn/data-science-machine-learning/natural-language-processing-with-deep-learning_293)



# MÔ HÌNH GOOGLE BERT



## I. Scaled-Dot Product & Multi-head Attention

## II. BERT

## III. BERT Objective

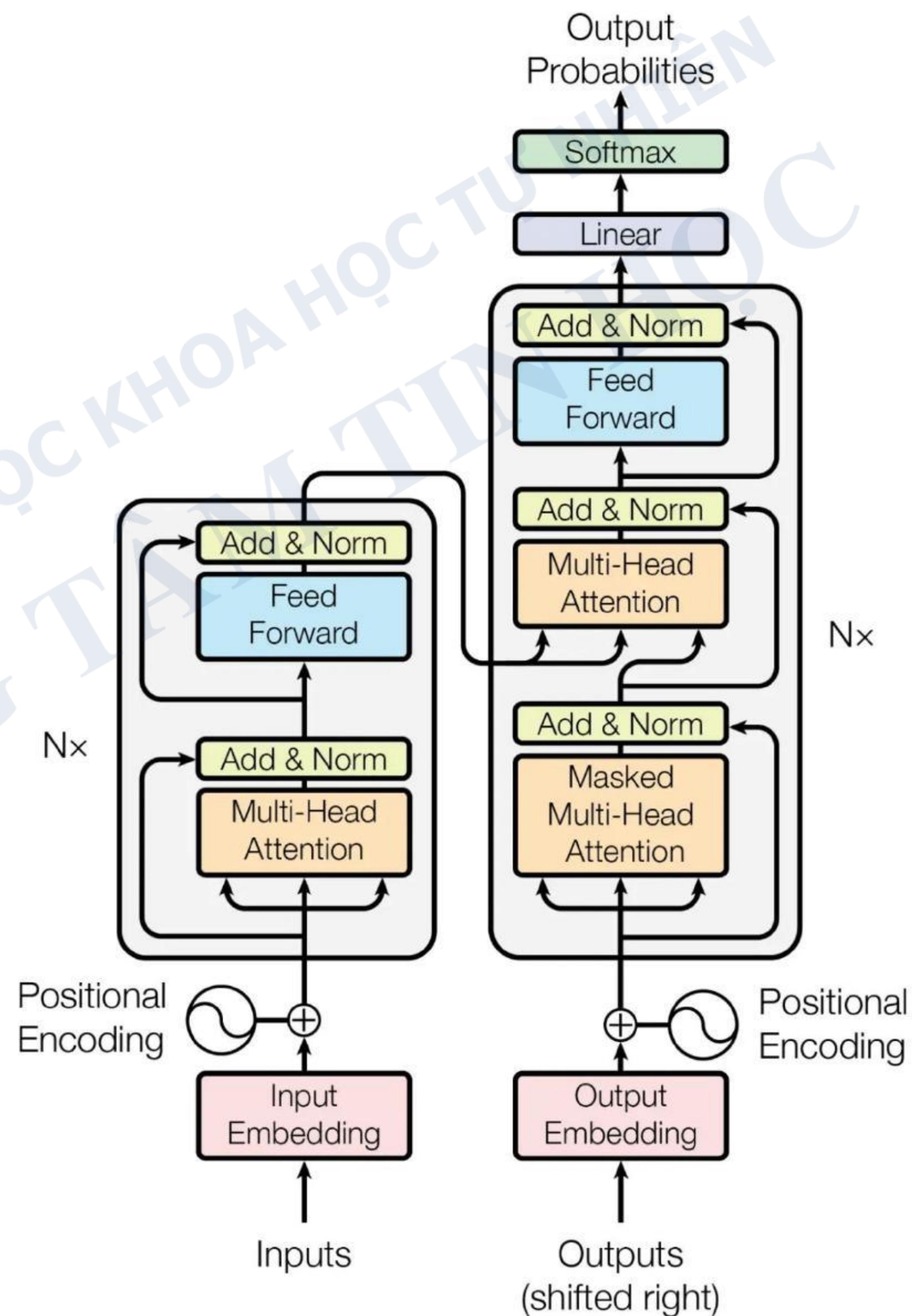
## IV. Subword Tokenization

## V. Finetuning BERT

# Scaled-Dot Product Attention

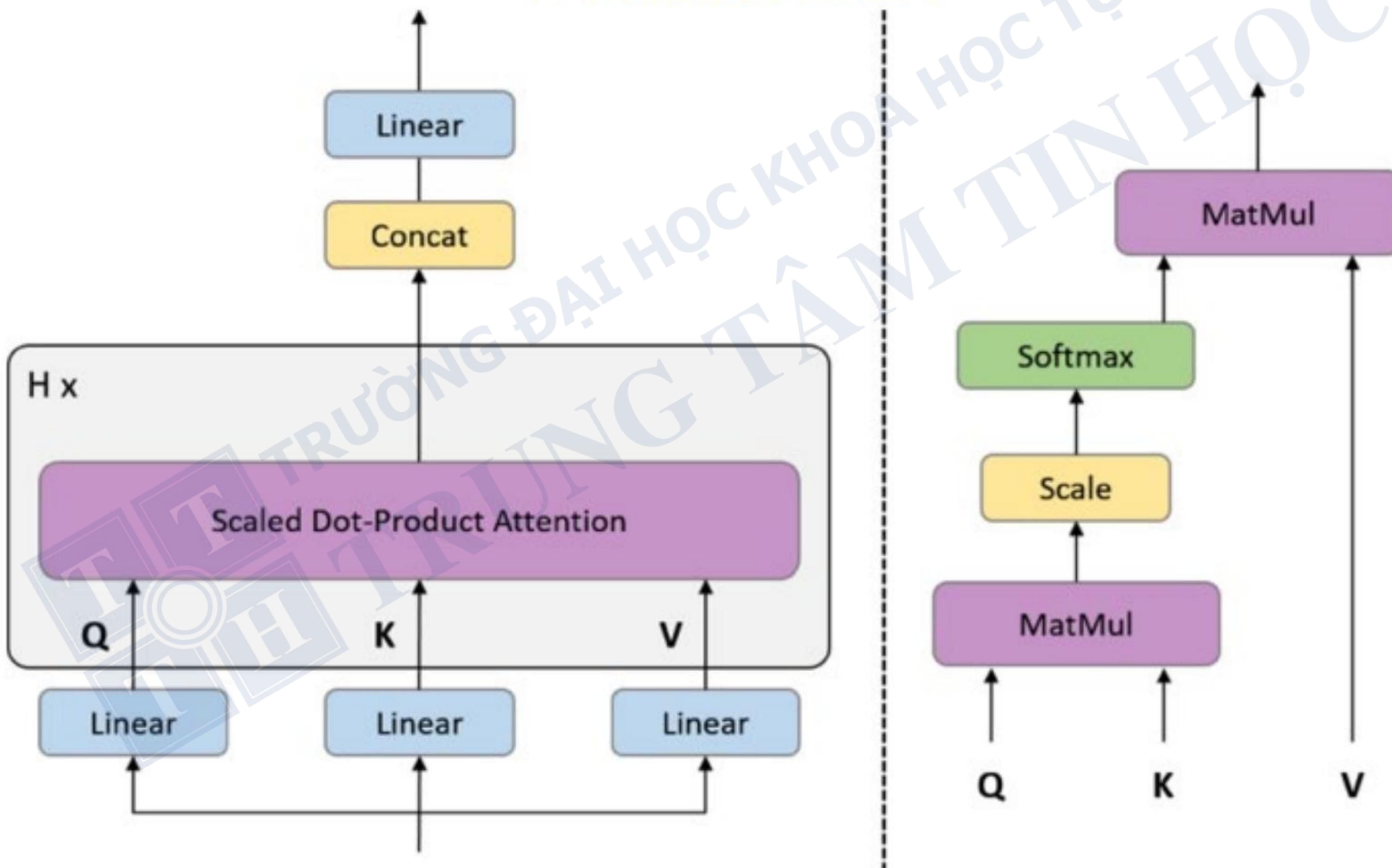
Là một phương pháp tính toán **attention weights** giữa các input trong mô hình transformer.

→ Input được nhân với một hệ số tỉ lệ nhằm điều chỉnh tỉ trọng của chúng.



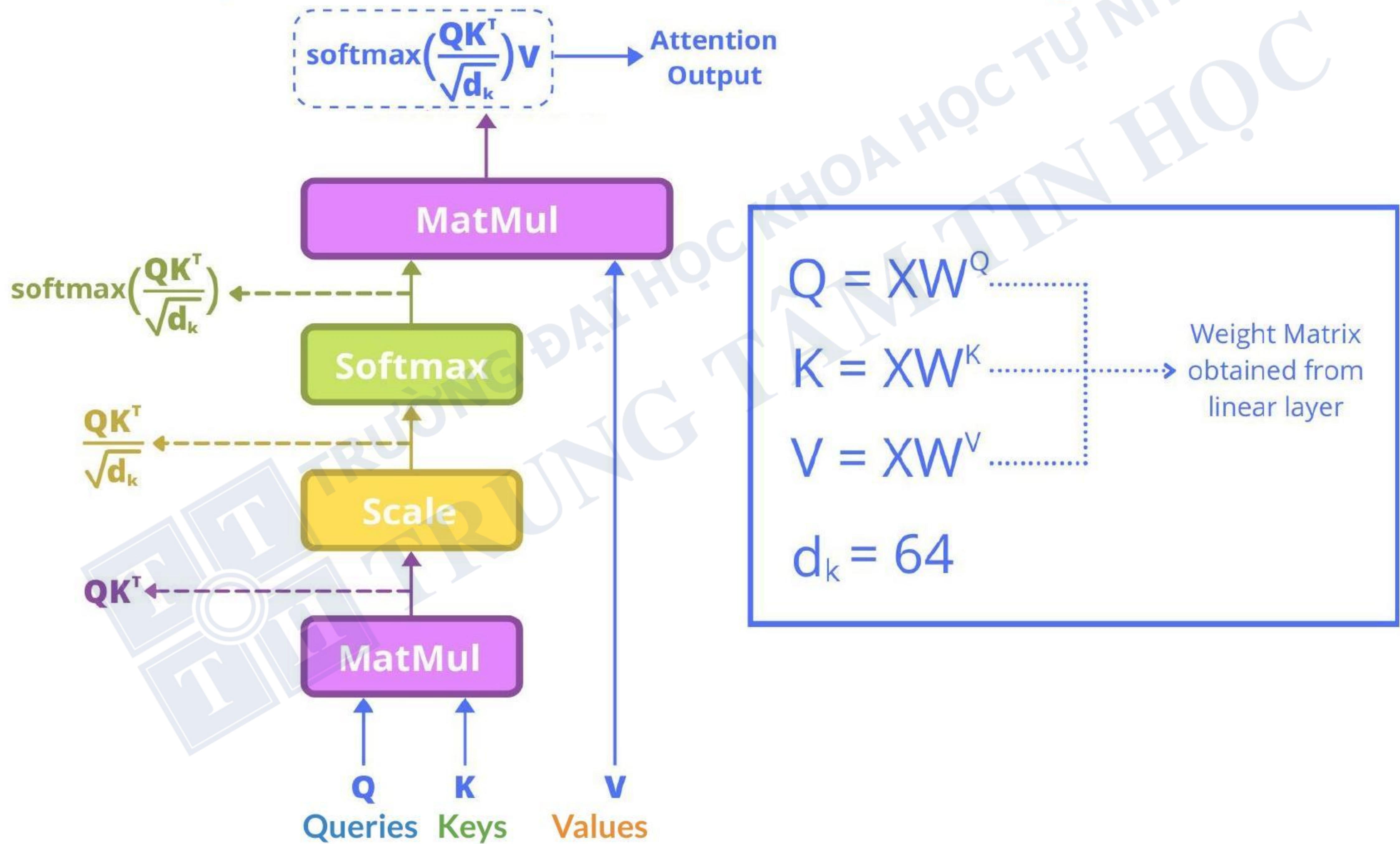
# Scaled-Dot Product Attention

Scaled-dot product attention trong mô hình Transformer.



# Scaled-Dot Product Attention

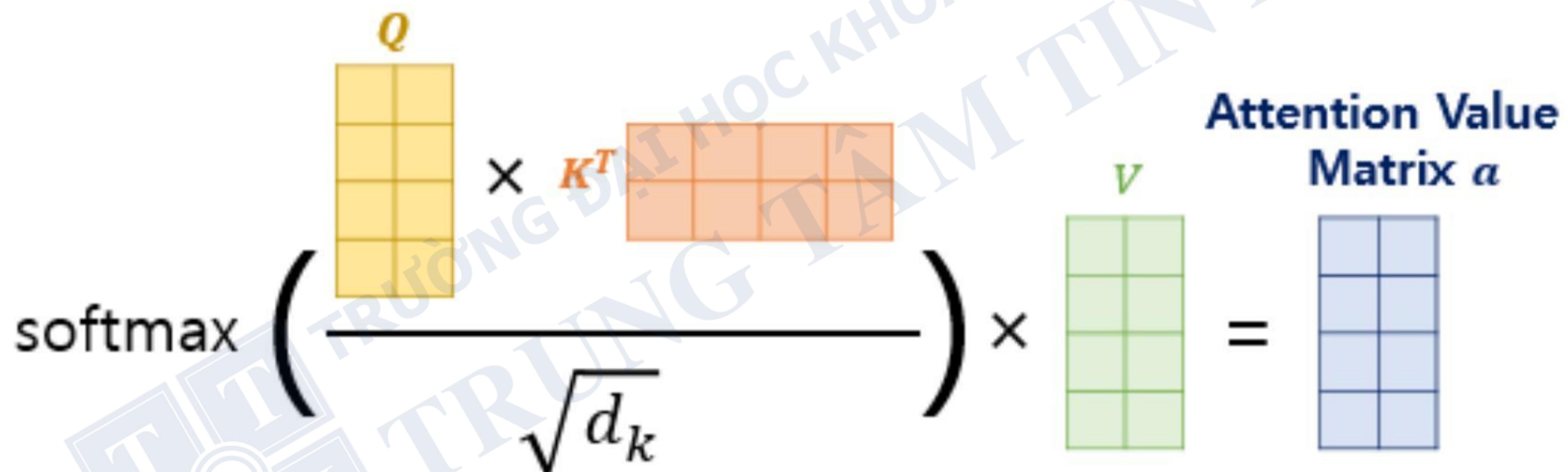
Có input được biểu diễn dưới dạng ma trận.





# Scaled-Dot Product Attention

Scale-dot Attention tính toán attention weights bằng cách lấy tích vô hướng giữa các hàng của ma trận input.



Sau đó chia cho một hệ số tỉ lệ để điều chỉnh tỷ trọng của các input.



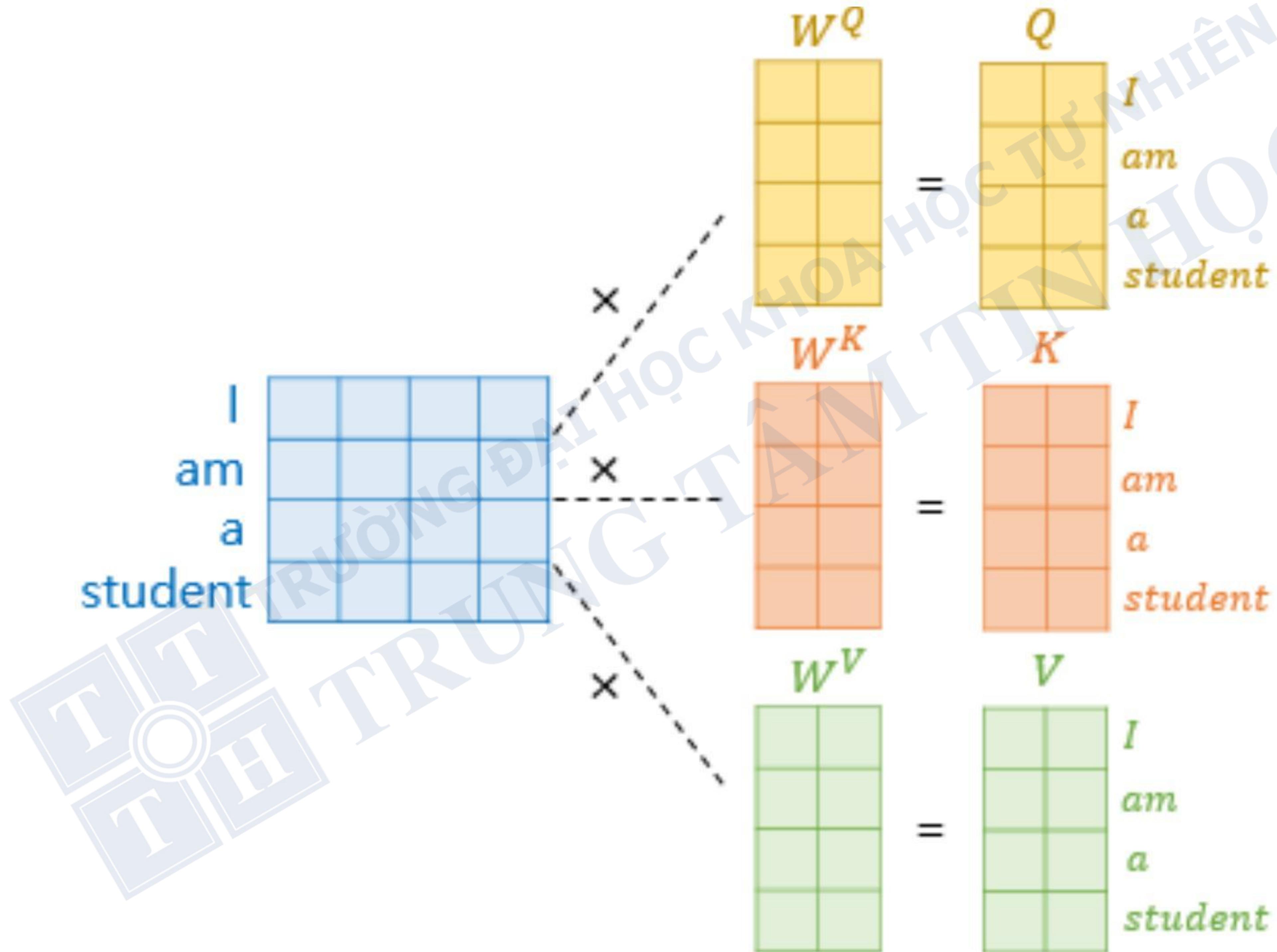
## Chức năng của Scaled-Dot Product Attention

Giúp mô hình transformer tập trung vào các input quan trọng hơn.

Giảm ảnh hưởng của các input không quan trọng.

- Cải thiện hiệu suất và khả năng biểu diễn của mô hình.
- Kiểm soát độ lớn của attention weights để tránh sự phụ thuộc quá mức vào một số input.

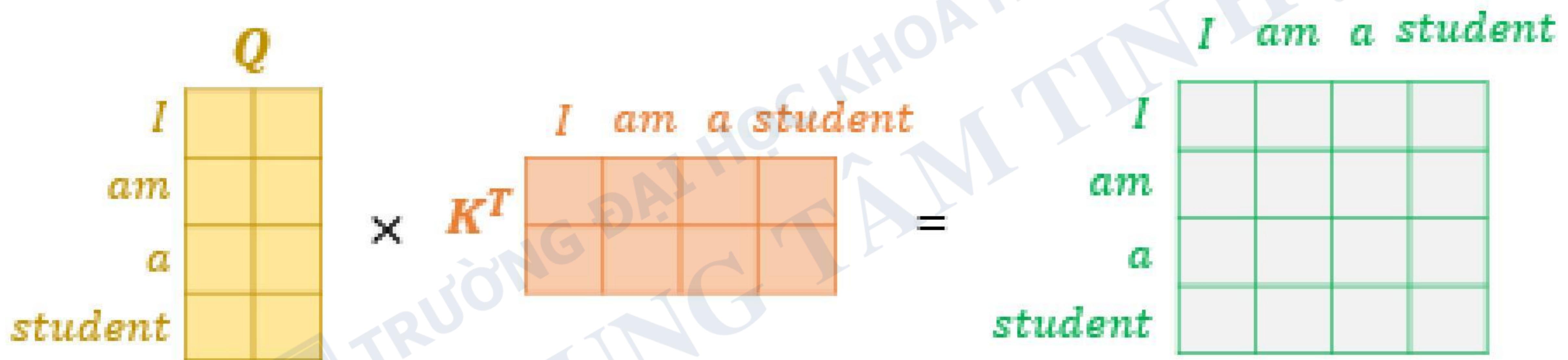
# Scaled-Dot Product Attention





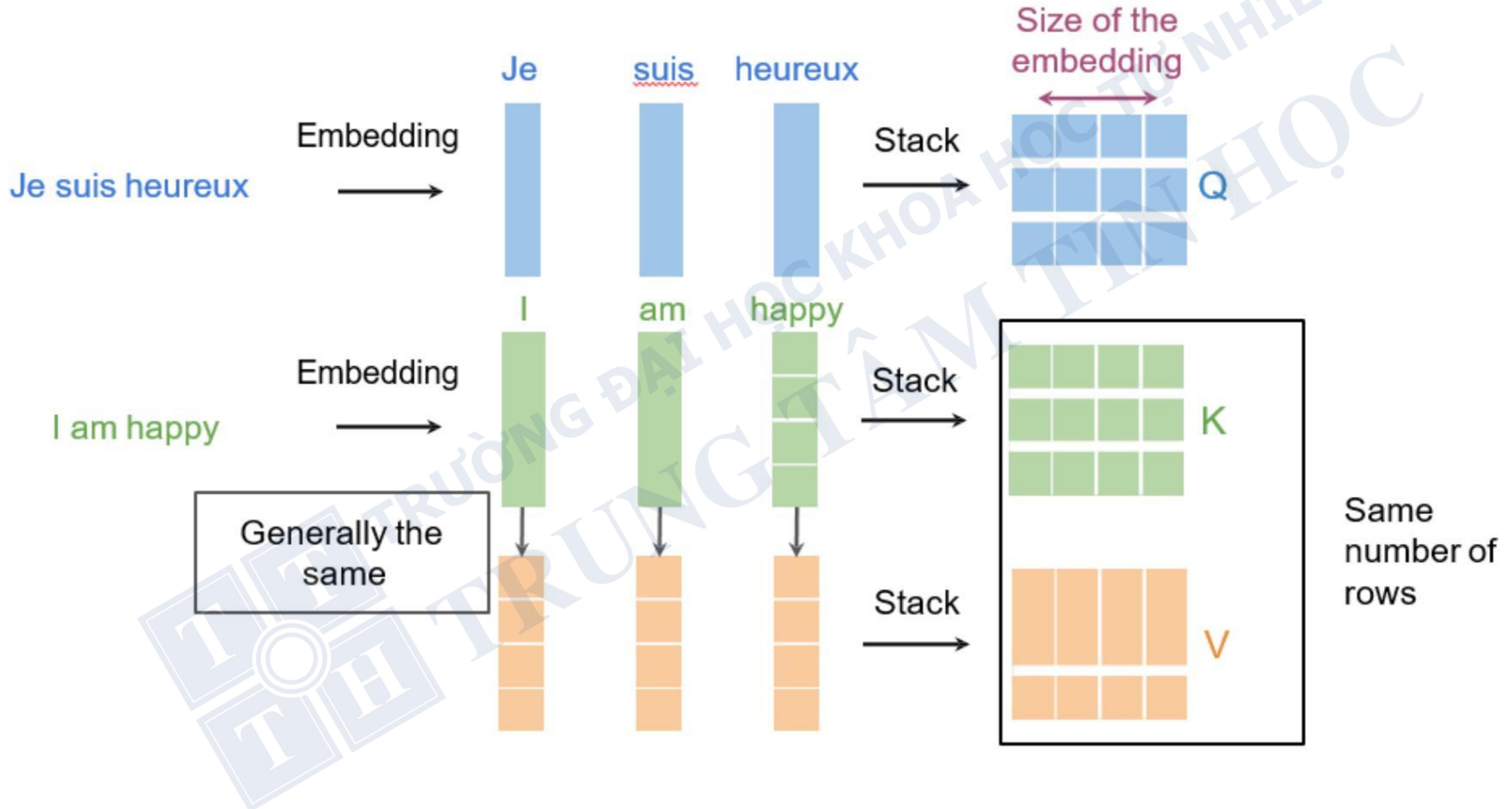
# Scaled-Dot Product Attention

Sử dụng công thức scale-dot attention để tính tỉ trọng của attention.

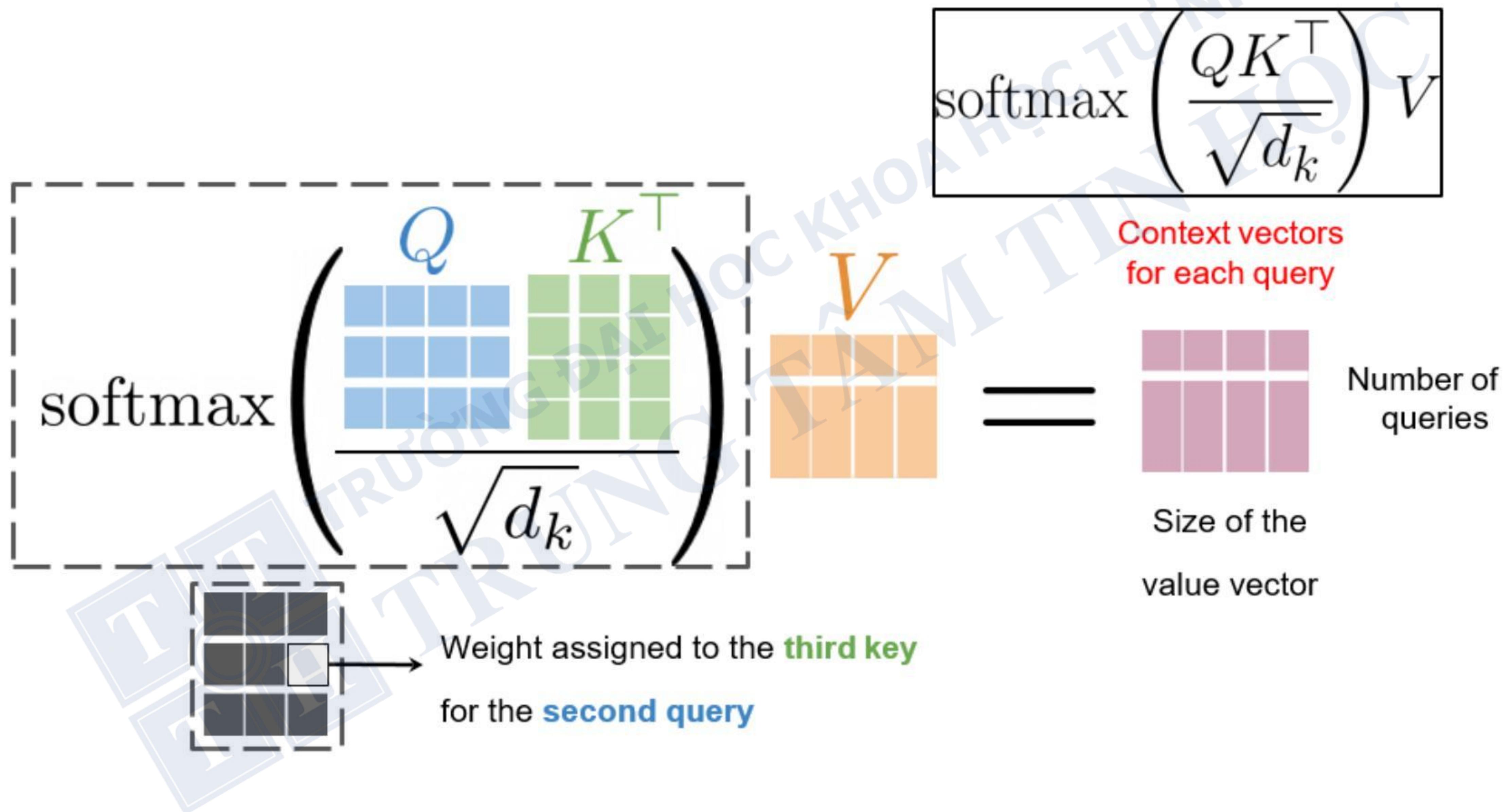


$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

# Scaled-Dot Product Attention

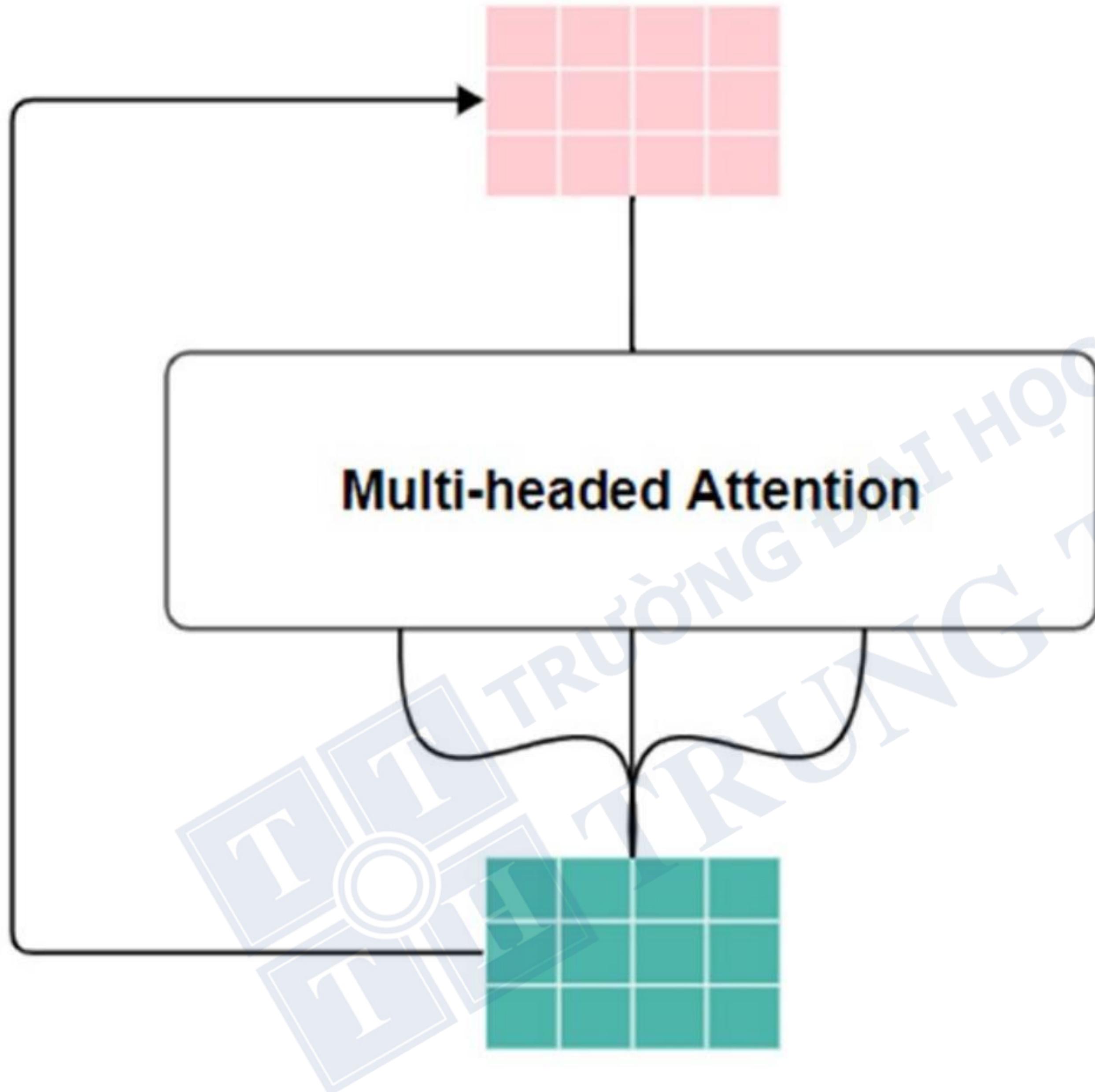


# Scaled-Dot Product Attention



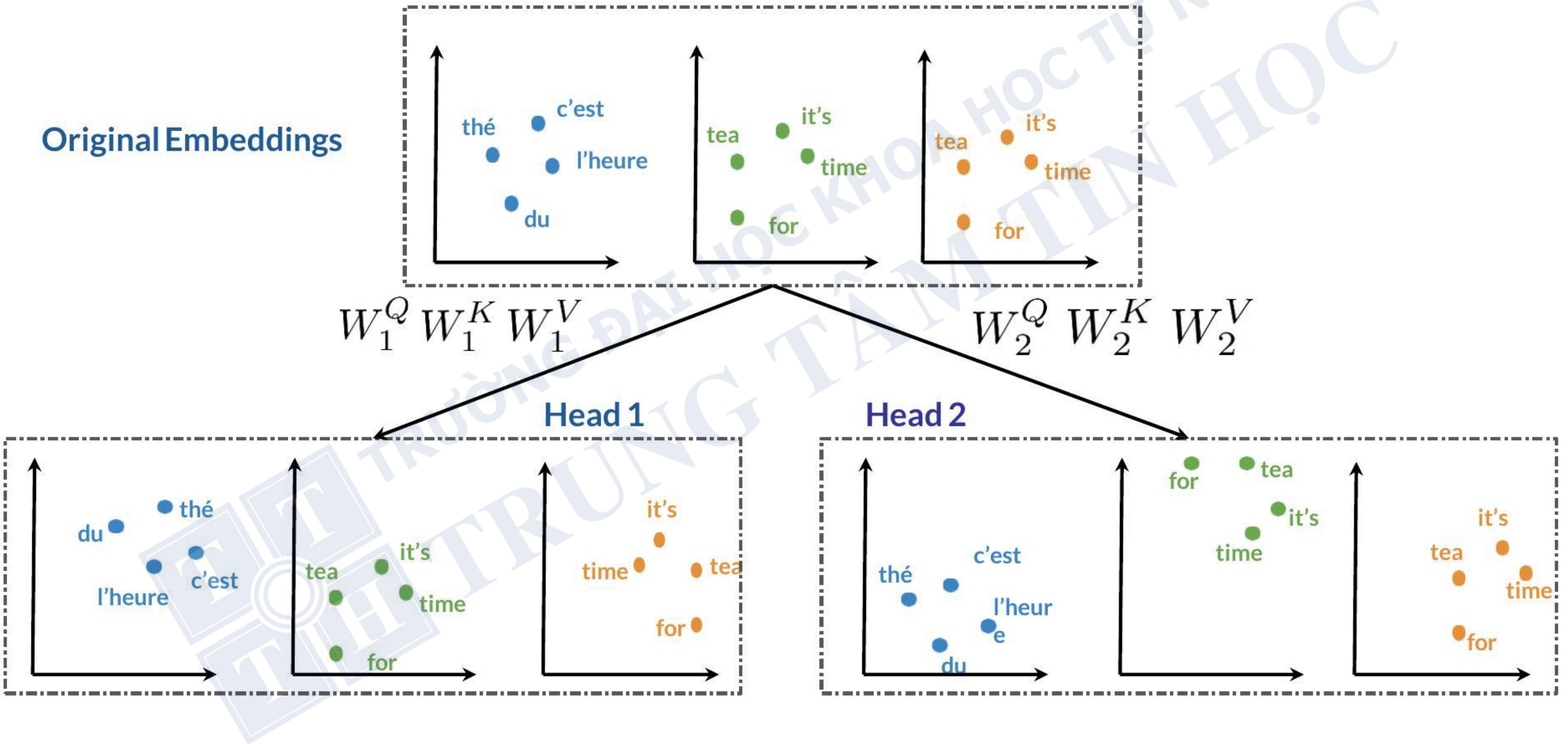


# Multi-Head Attention

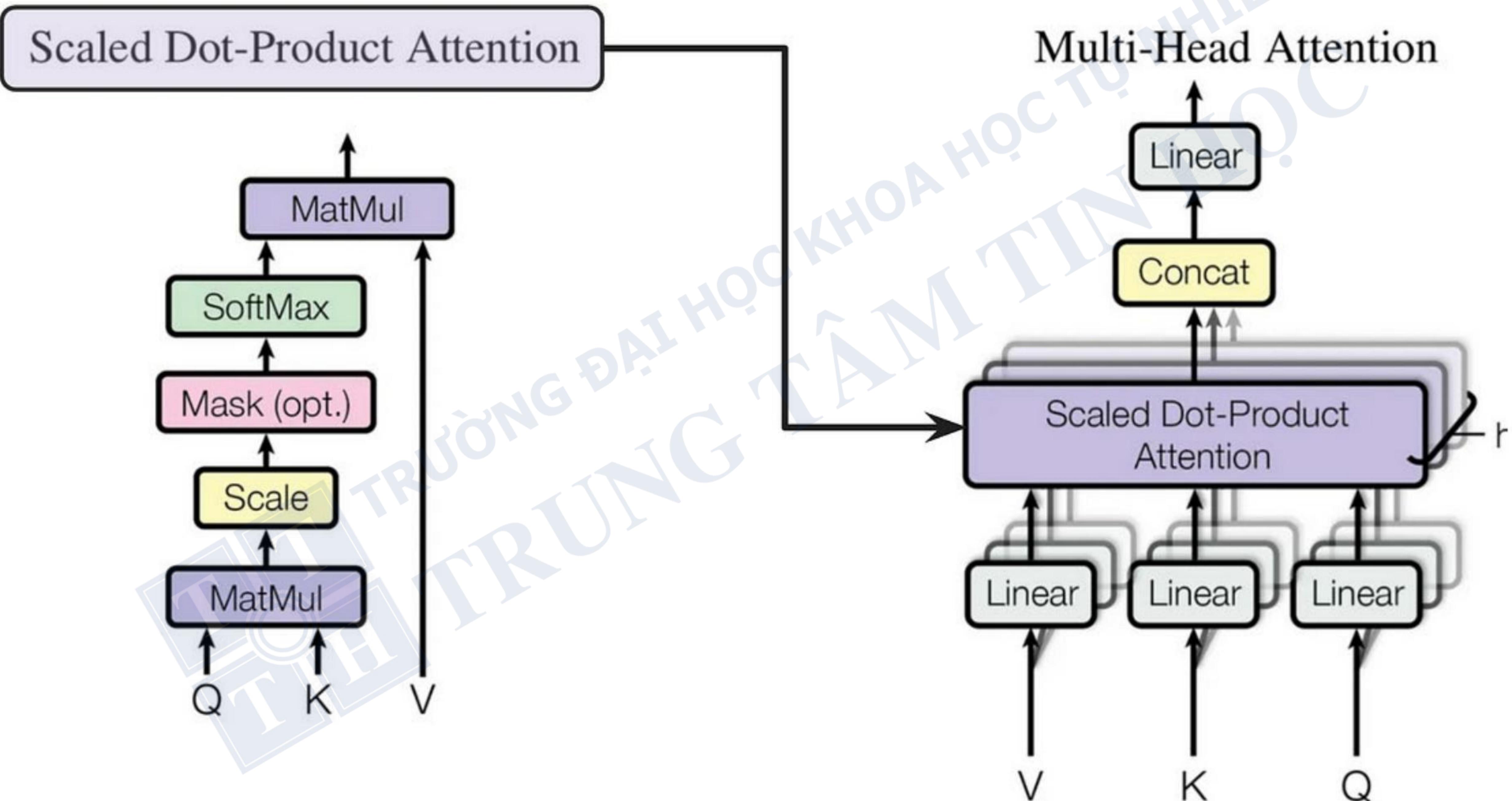


Là một phương pháp tính toán attention weights trong mô hình transformer như **scaled dot-product attention**, nhưng sử dụng nhiều head khác nhau (multi-head).

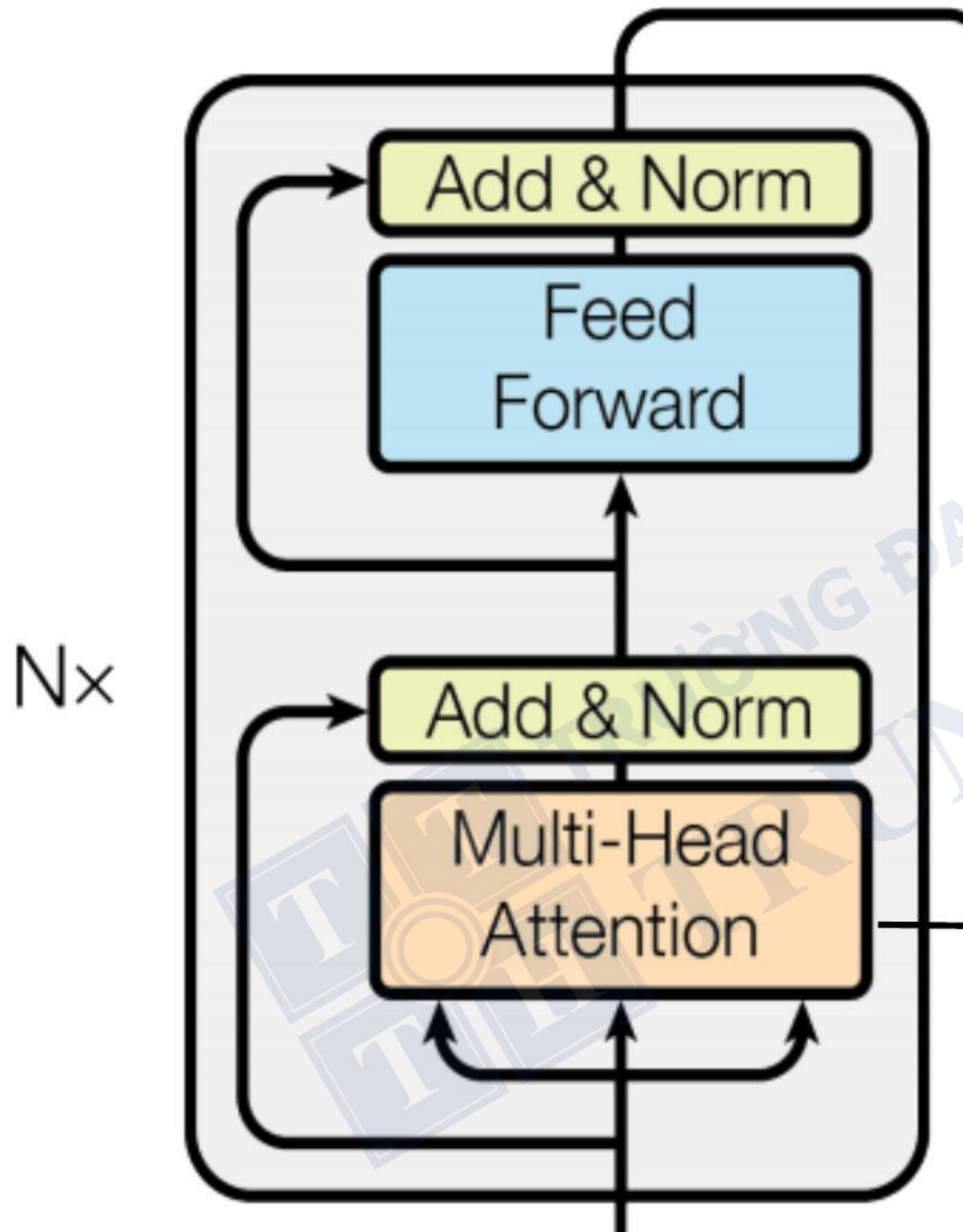
# Multi-Head Attention



# Multi-Head Attention

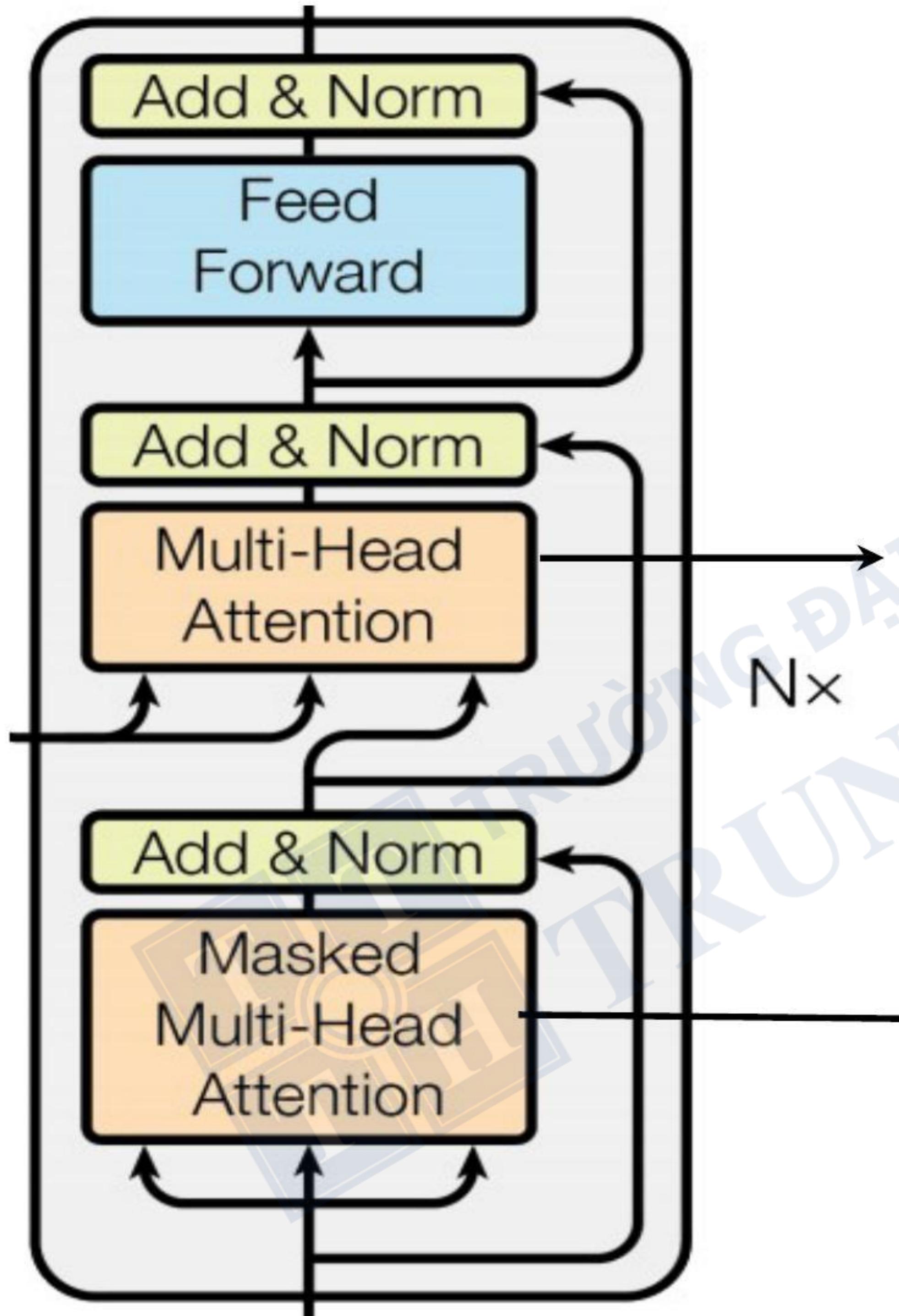


# Multi-Head Attention



1. Input được chia thành nhiều phần bằng cách áp dụng các **ma trận trọng số** khác nhau cho từng head.
2. Mỗi head tính toán attention weights riêng của mình bằng cách sử dụng ma trận trọng số tương ứng.
3. Output từ mỗi head kết hợp lại tạo ra output cuối cùng.

# Multi-Head Attention



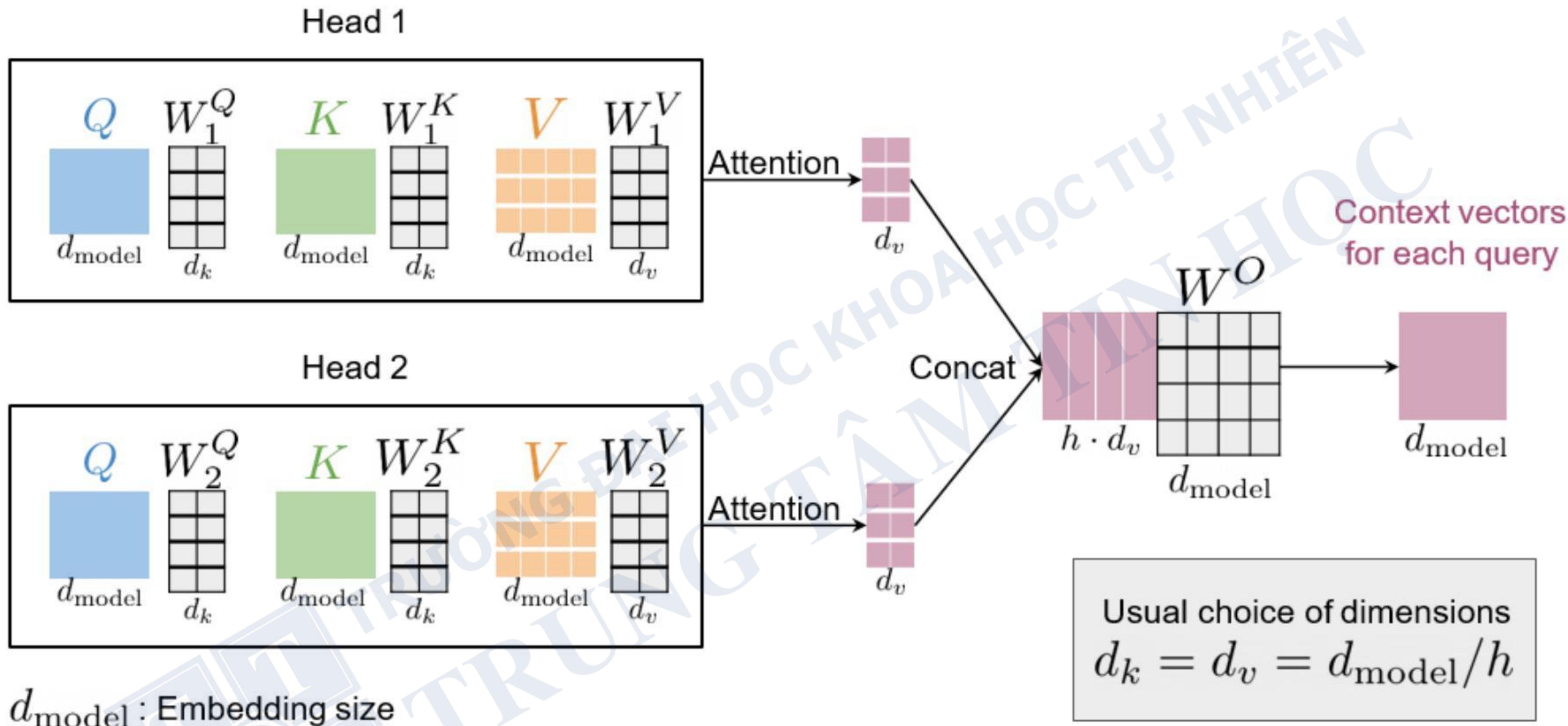
## Encoder-Decoder Attention

Every position from the decoder attends to the outputs from the encoder

## Masked Self-Attention

Every position attends to previous positions

# Multi-Head Attention



- Nhận thông tin từ các representation khác nhau.
- Tính toán attention song song.
- Chi phí tính toán tương tự như **single-head attention**.

# MÔ HÌNH GOOGLE BERT



## I. Scaled-Dot Product & Multi-head Attention

## II. BERT

## III. BERT Objective

## IV. Sub word Tokenization

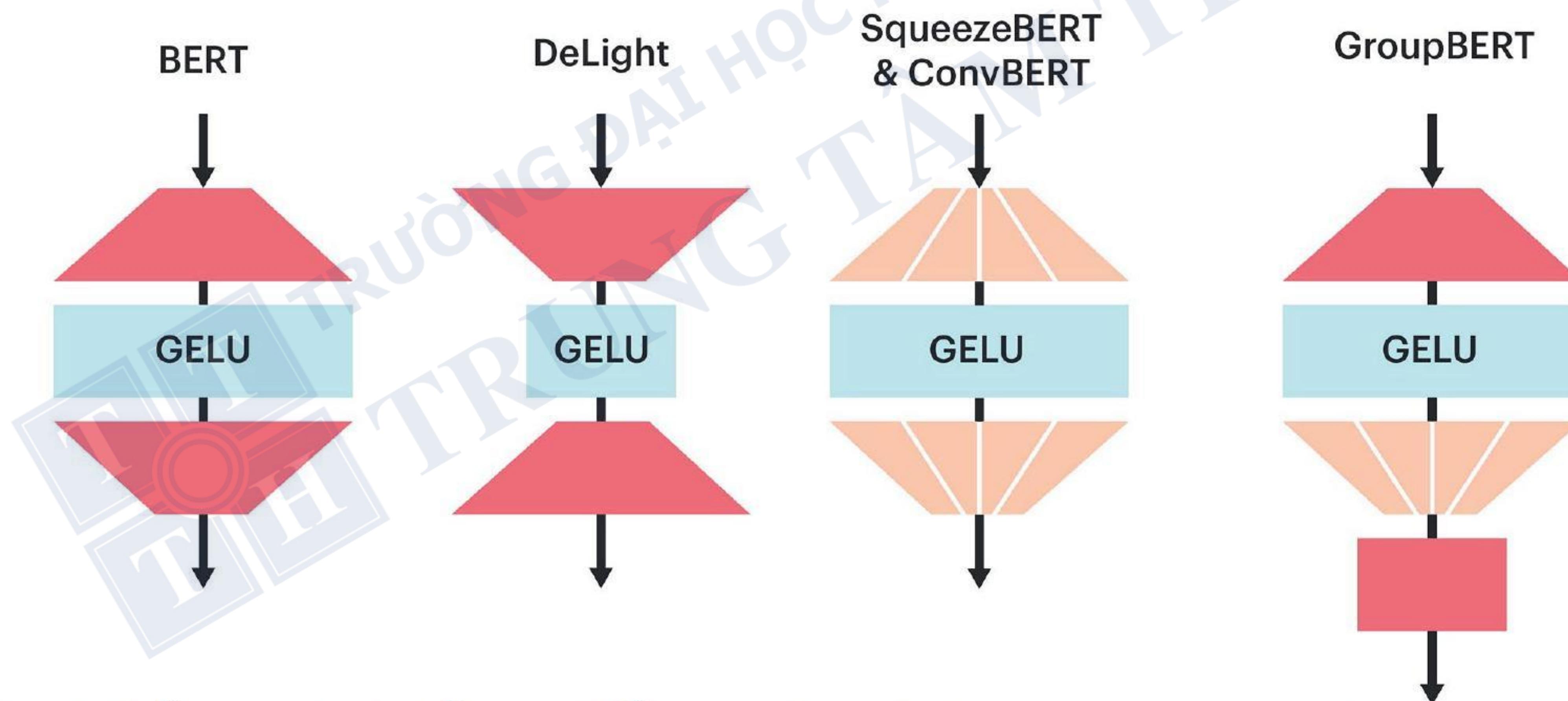
## V. Finetuning BERT



# BERT Model

*Bidirectional Encoder Representations from Transformers* là một mô hình ngôn ngữ tự nhiên dựa trên transformer.

Được train với bộ dữ liệu ngôn ngữ tự nhiên lớn.



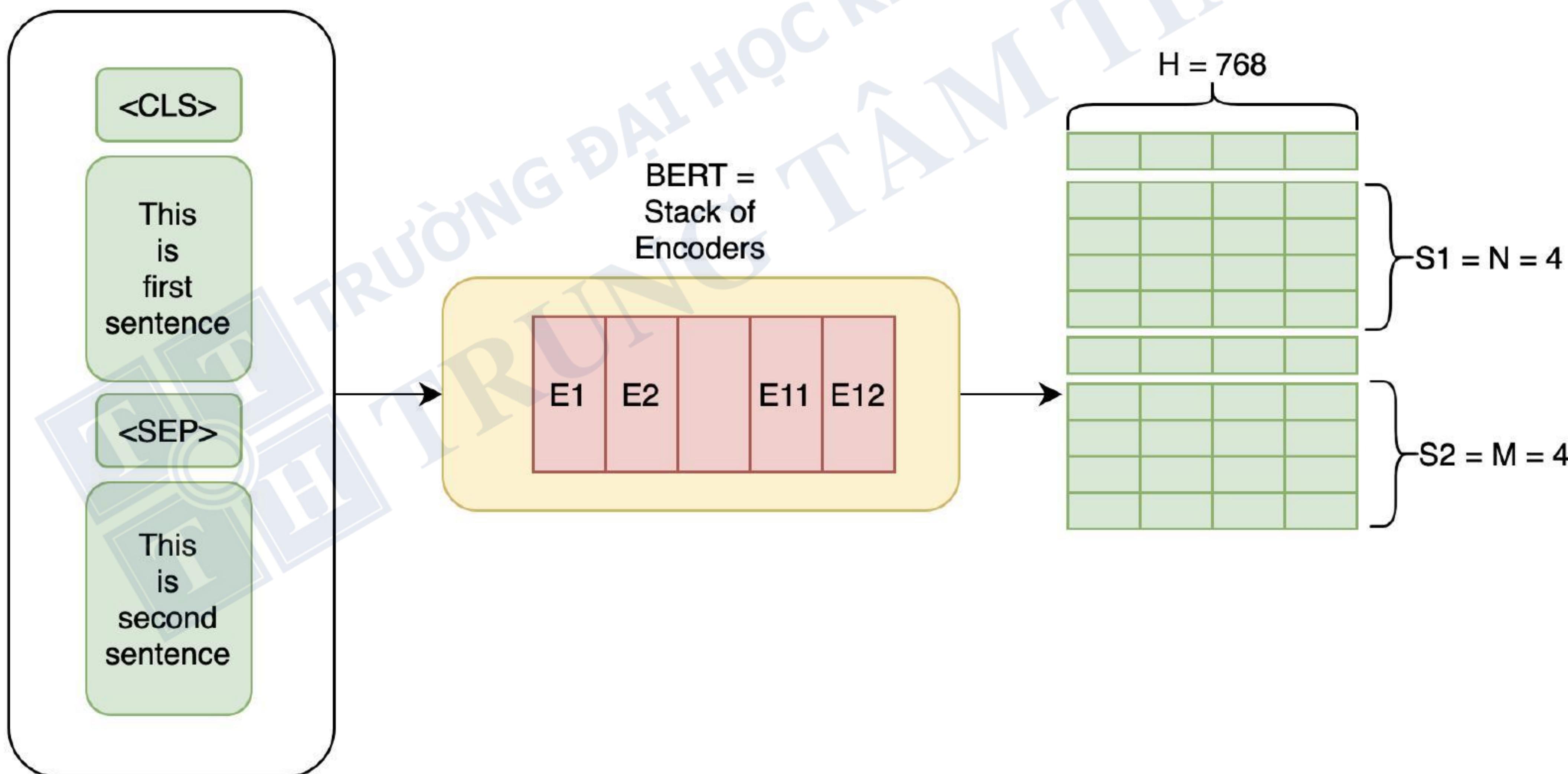
→ Hiểu và biểu diễn các đặc trưng ngôn ngữ.



# Kiến trúc của BERT

BERT sử dụng cấu trúc transformer với một encoder stack.

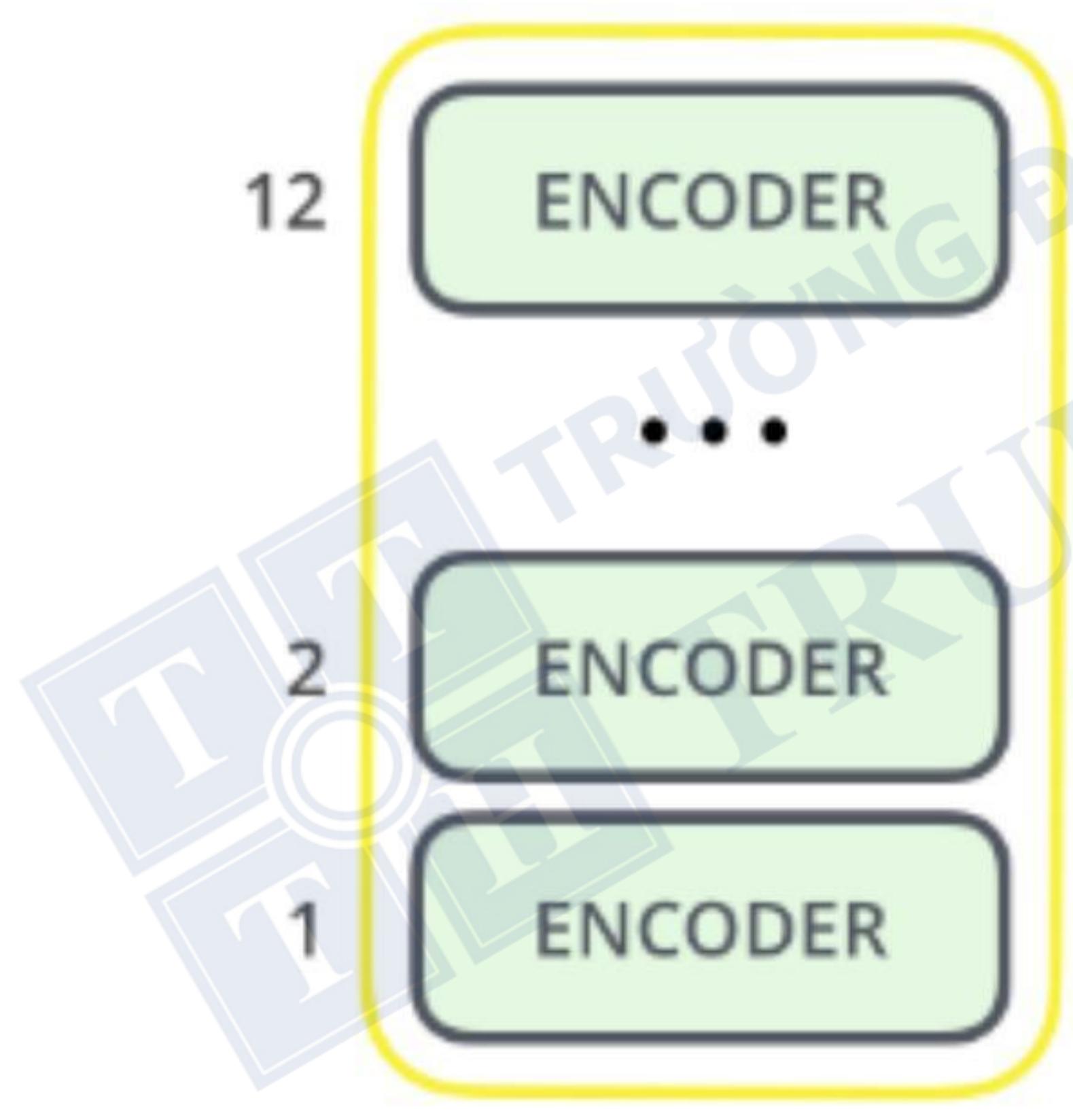
→ Mỗi encoder layer trong stack gồm một **self-attention mechanism** và một **feed-forward neural network**.



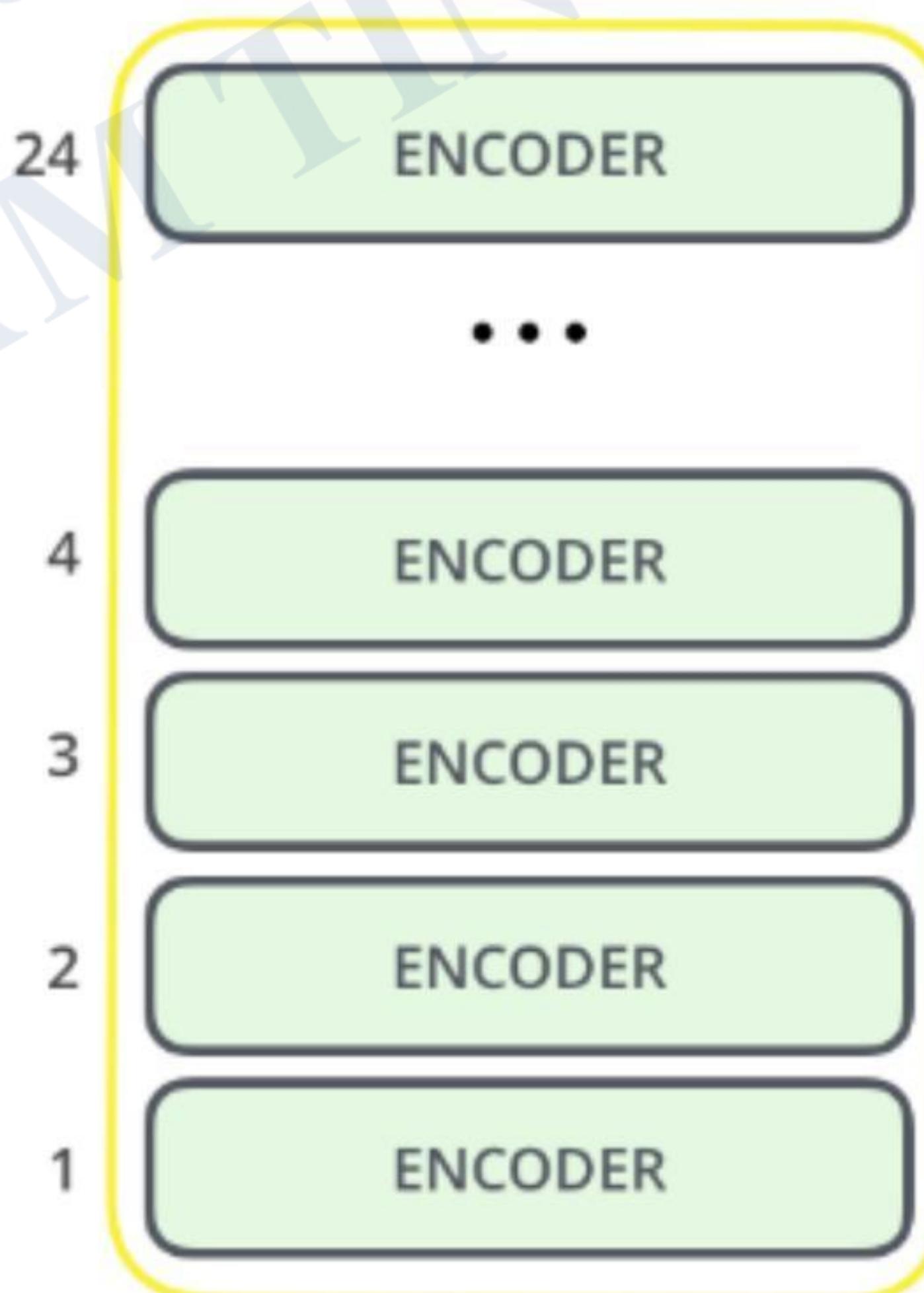


# Ưu điểm của BERT

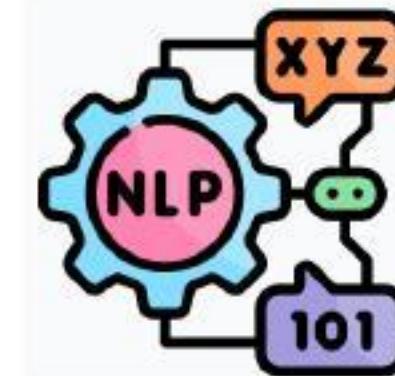
- Hiểu và biểu diễn các mối quan hệ ngữ nghĩa phức tạp.
- Xử lý ngữ cảnh và hiểu được các đặc trưng ngôn ngữ phụ thuộc vào ngữ cảnh.



BERT<sub>BASE</sub>



BERT<sub>LARGE</sub>



# Ứng dụng của BERT

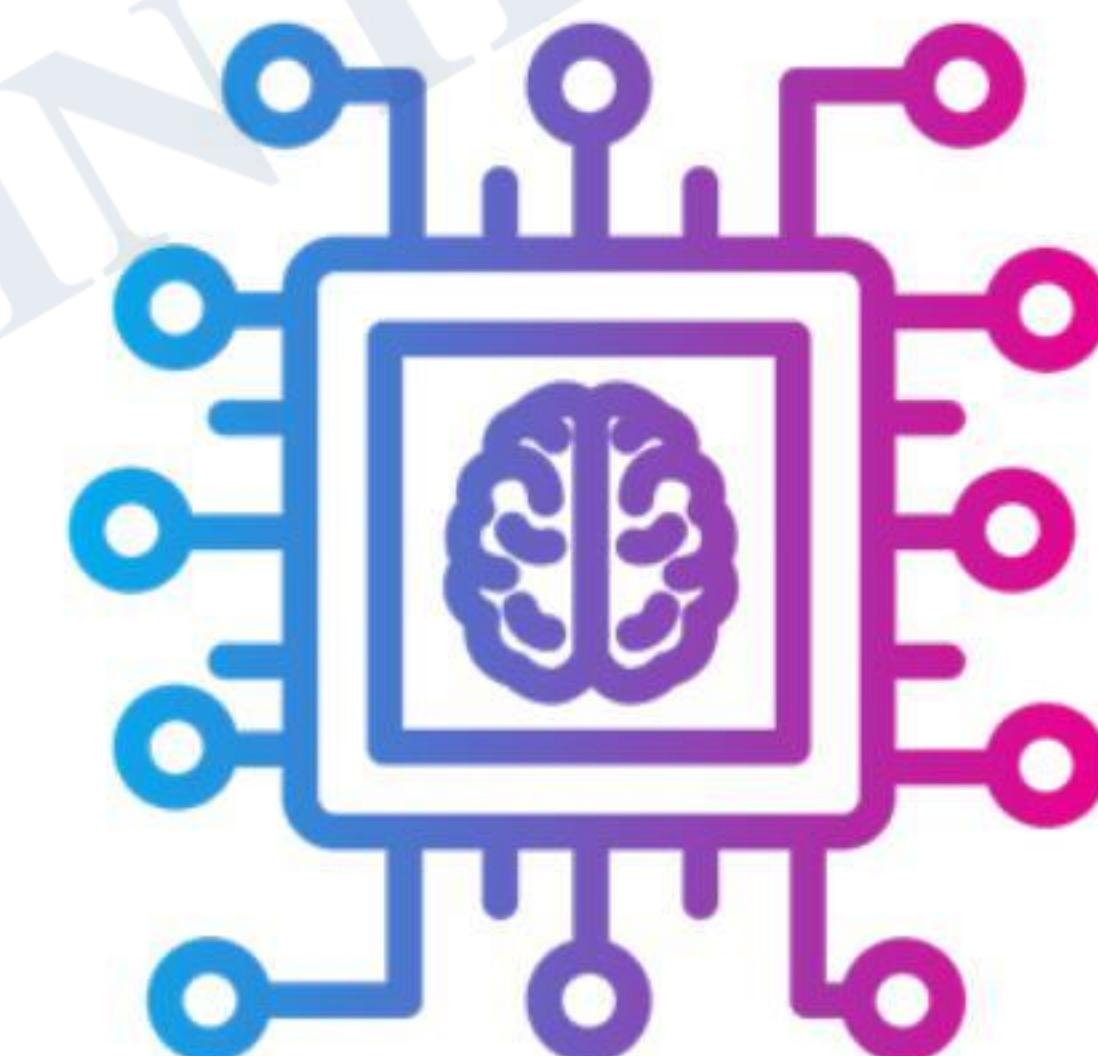
BERT có thể được sử dụng trong nhiều tác vụ liên quan đến xử lý ngôn ngữ tự nhiên như:



Dịch máy



Phân loại văn bản



Trích xuất thông tin

→ Là một trong những mô hình quan trọng và phổ biến nhất trong xử lý ngôn ngữ tự nhiên.

# MÔ HÌNH GOOGLE BERT



## I. Scaled-Dot Product & Multi-head Attention

## II. BERT

## III. BERT Objective

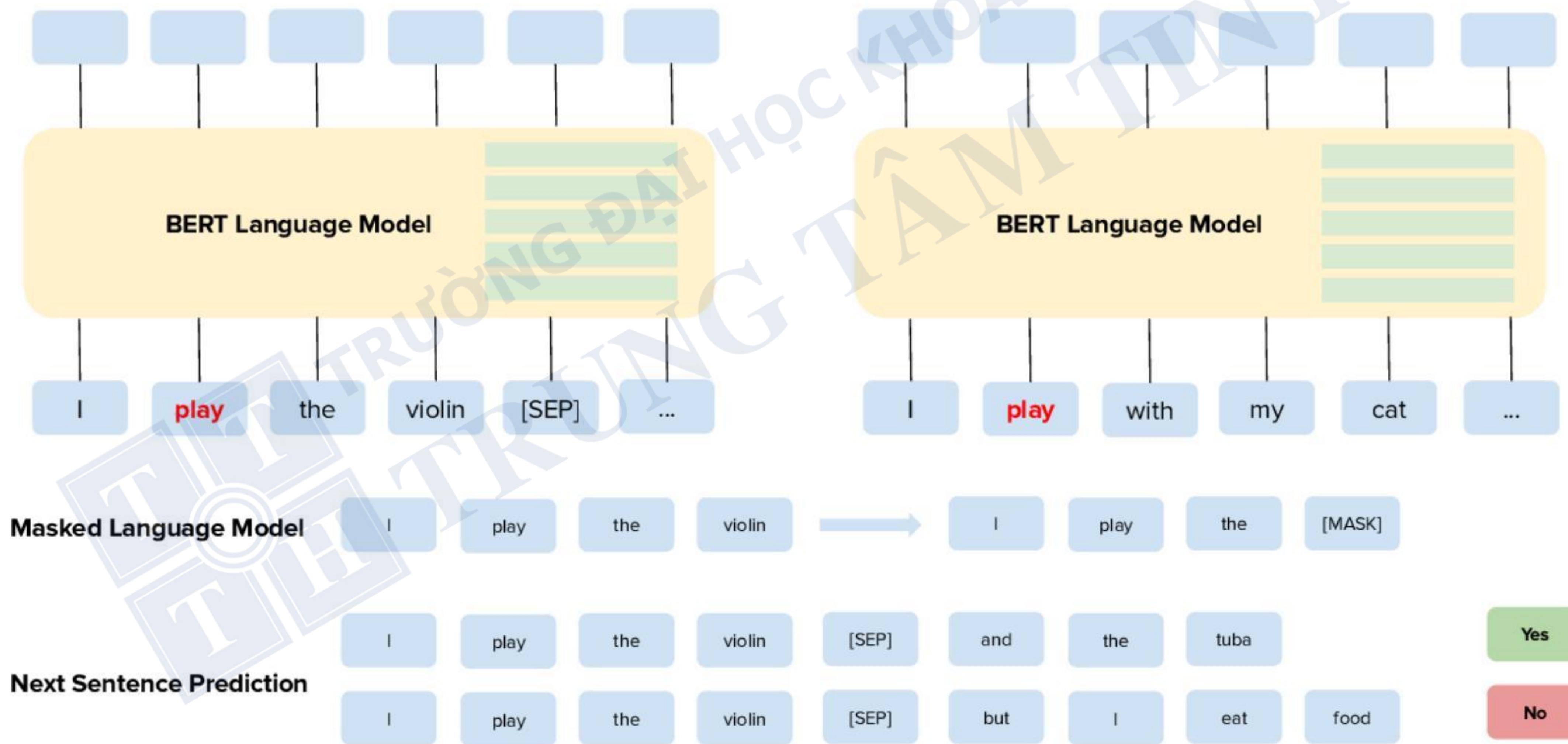
## IV. Subword Tokenization

## V. Finetuning BERT

# BERT Objective

BERT Objective bao gồm hai tác vụ chính:

- **Masked Language Model (MLM)**
- **Next Sentence Prediction (NSP)**.



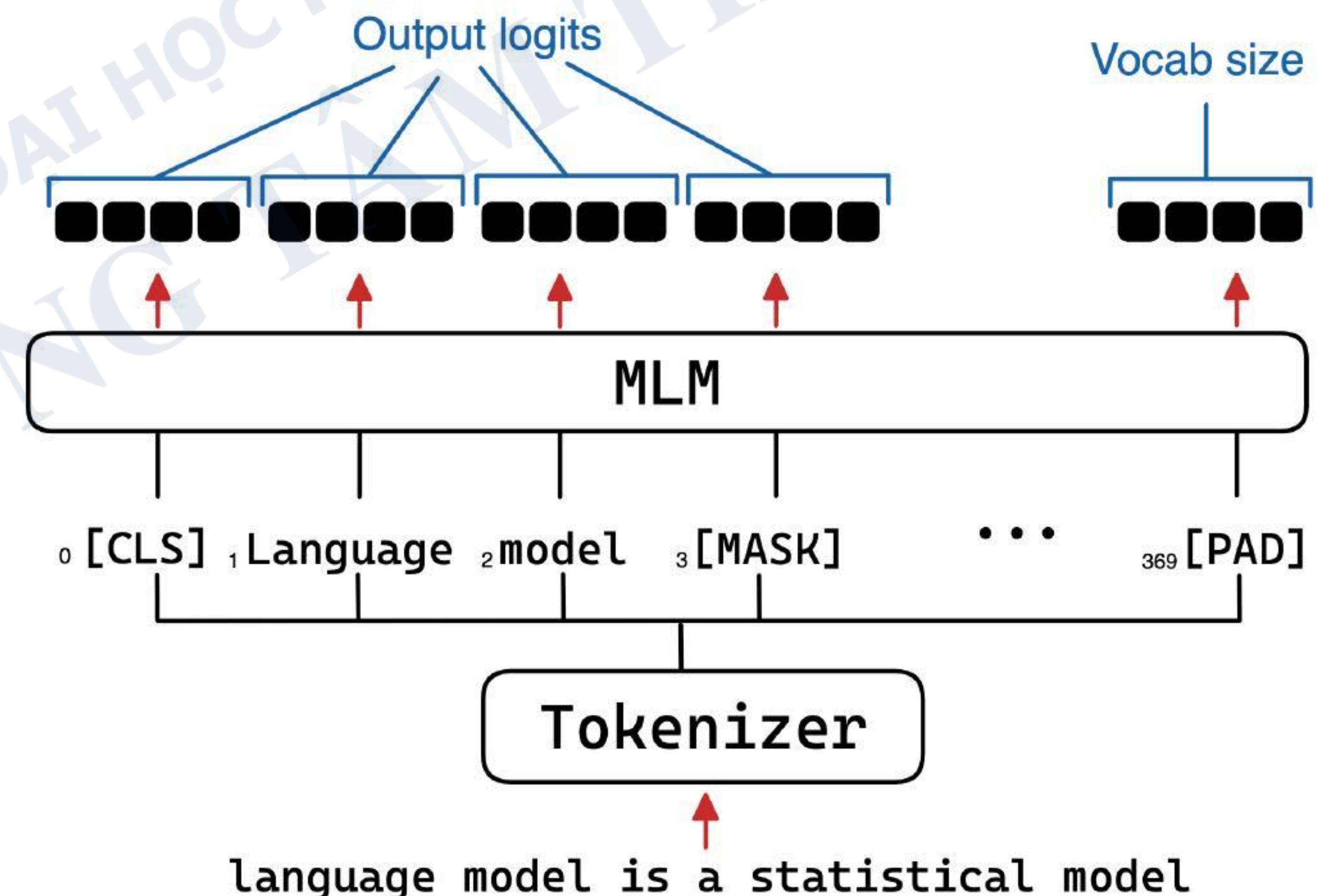


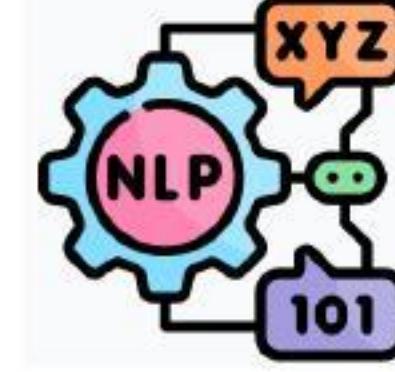
# BERT Objective

## 1. Masked Language Model (MLM)

- Một số từ trong câu được ngẫu nhiên che đi.
- Mô hình BERT cố gắng dự đoán các từ bị che dựa trên ngữ cảnh xung quanh.

→ Giúp mô hình hiểu và biểu diễn các **mối quan hệ ngữ nghĩa** trong câu.



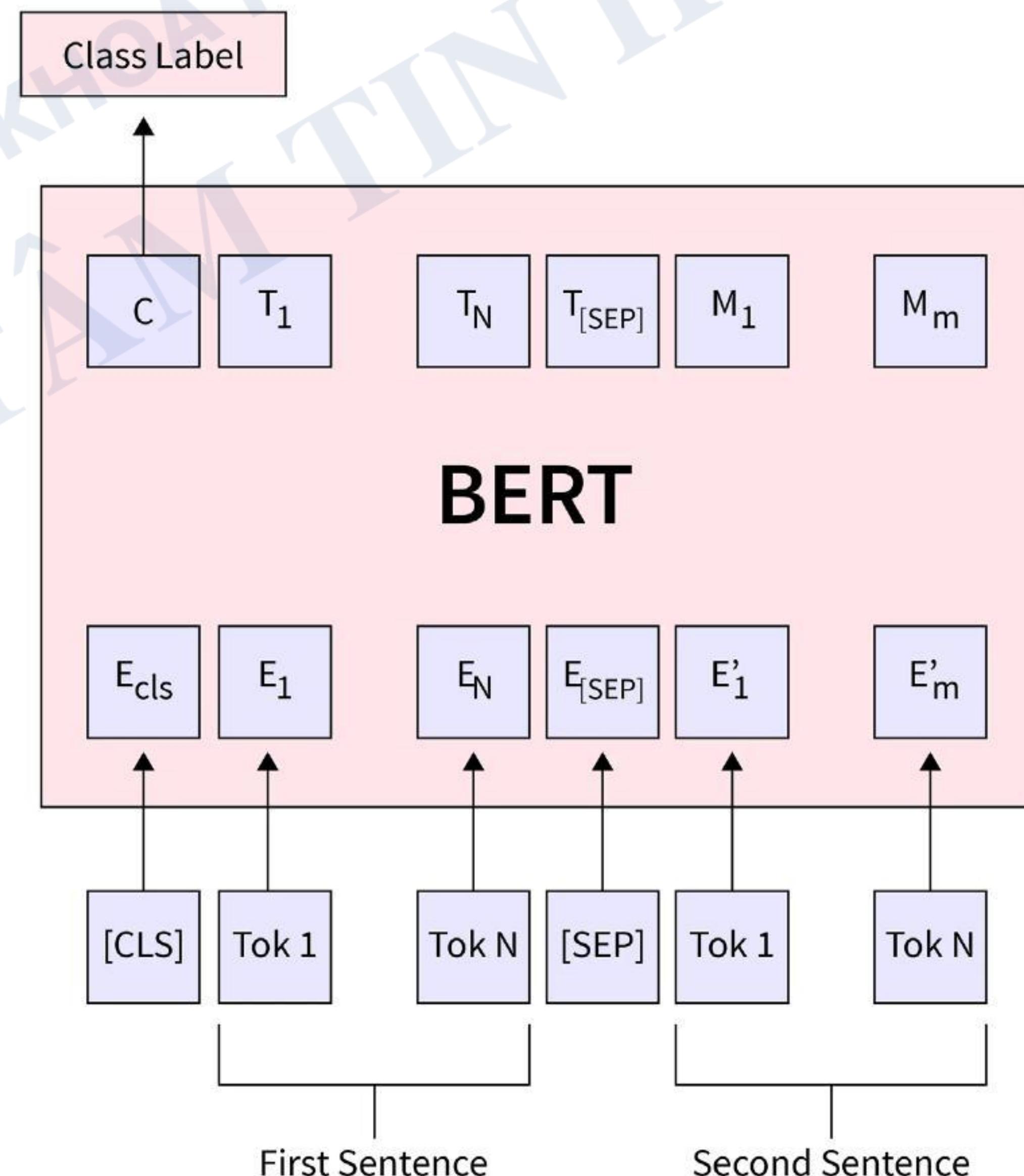


# BERT Objective

## 1. Next Sentence Prediction (NSP):

- Hai câu được chọn ngẫu nhiên từ training set.
- Mô hình BERT cố gắng dự đoán xem hai câu có liên quan nhau hay không.

→ Giúp mô hình hiểu  
được **sự tương quan**  
giữa các câu trong ngữ  
cảnh tổng thể.



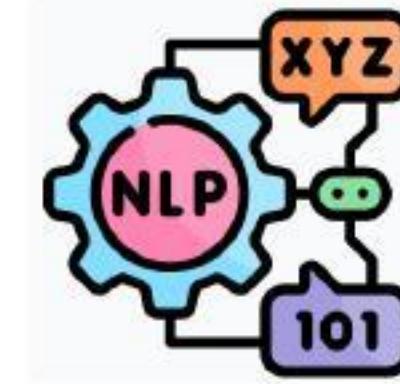


# BERT Objective

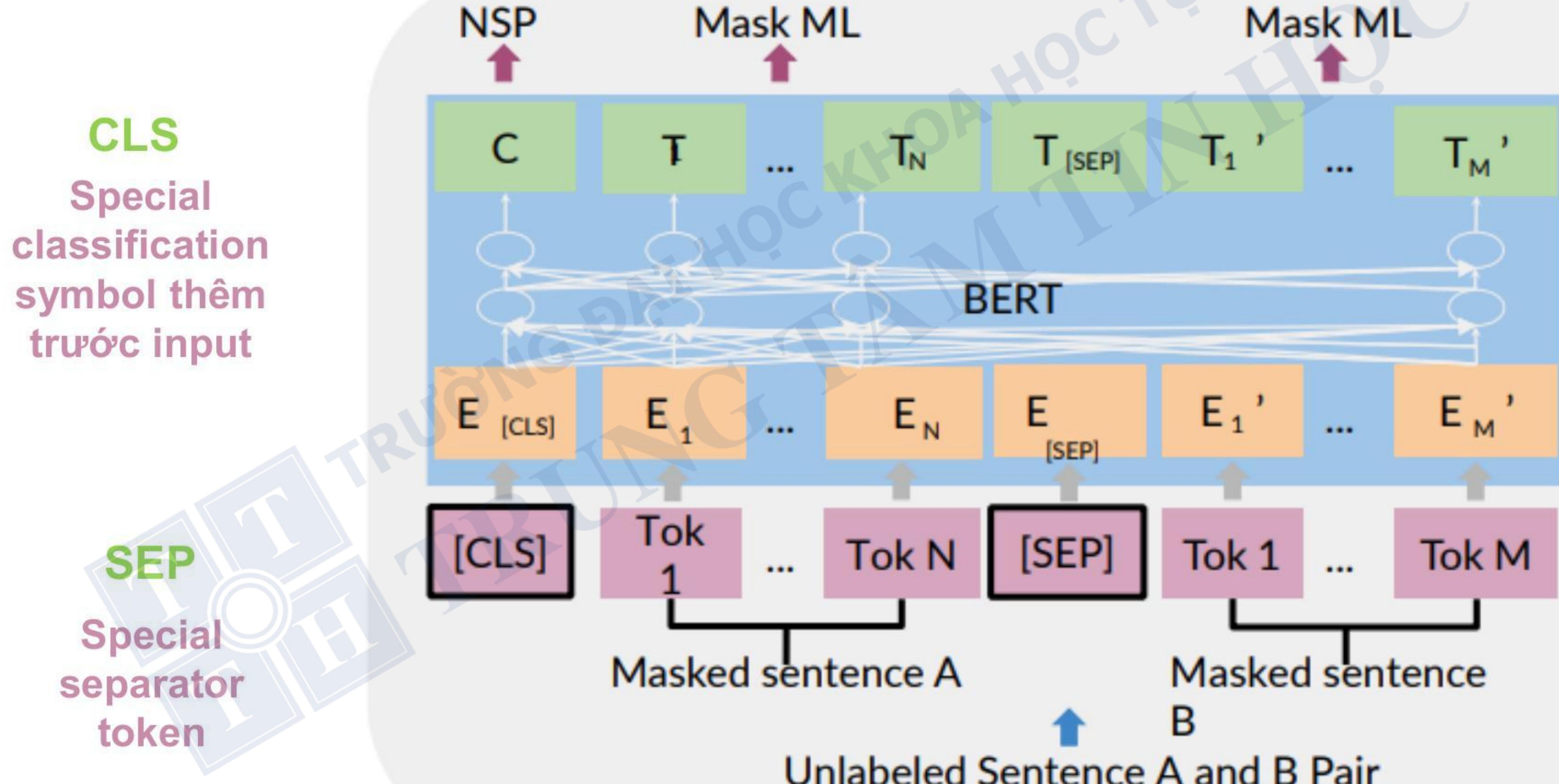
## Formalizing the input

Input	[CLS]	my	dog	is	cute	[SEP]	he	likes	play	##ing	[SEP]
Token Embeddings	E [CLS]	E my	E dog	E is	E cute	E [SEP]	E he	E likes	E play	E ##ing	E [SEP]
Segment Embeddings	+	+	+	+	+	+	+	+	+	+	+
Position Embeddings	E A	E A	E A	E A	E A	E A	E B	E B	E B	E B	E B
	E 0	E 1	E 2	E 3	E 4	E 5	E 6	E 7	E 8	E 9	E 10

# BERT Objective

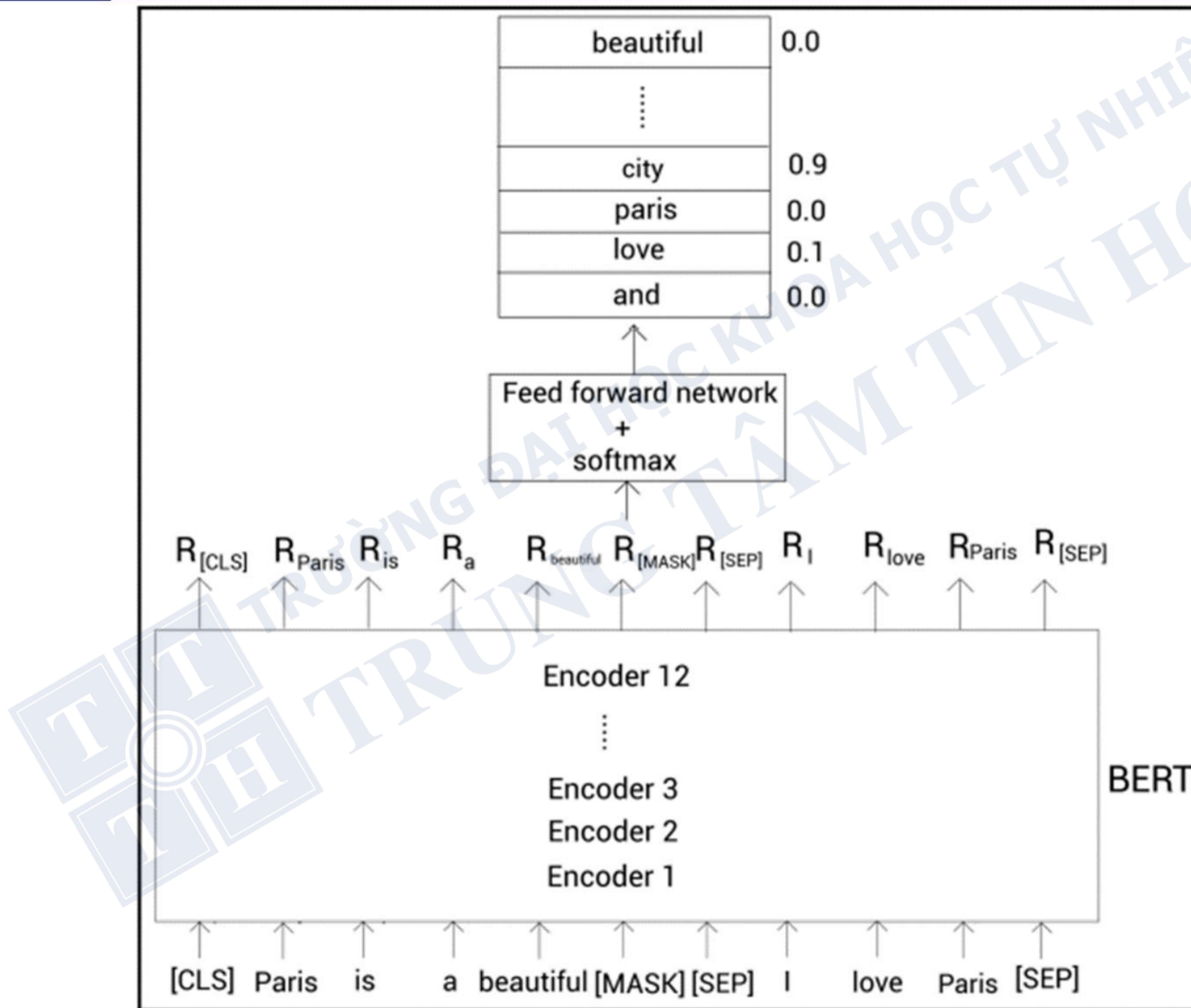


## Visualize the input

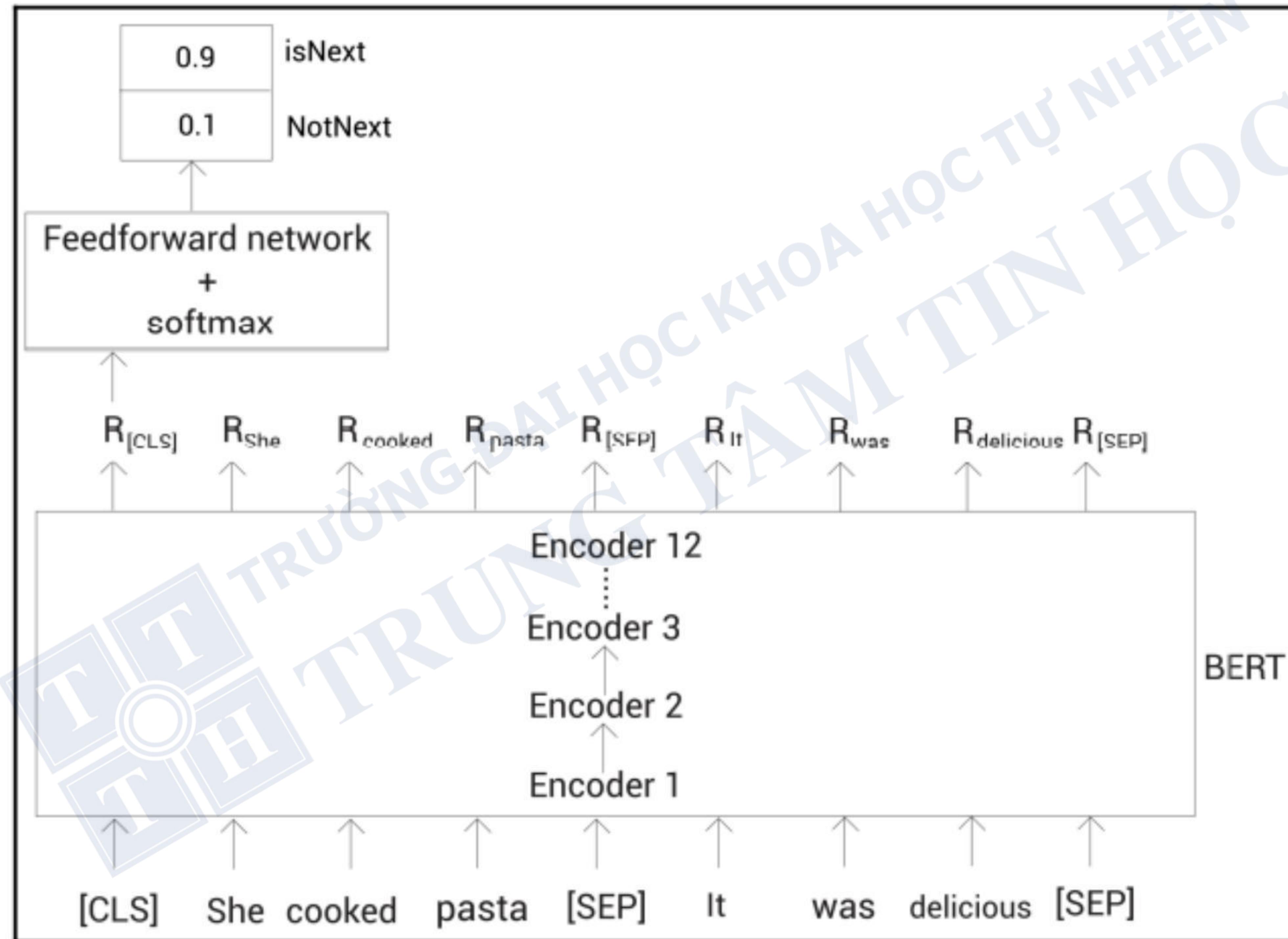




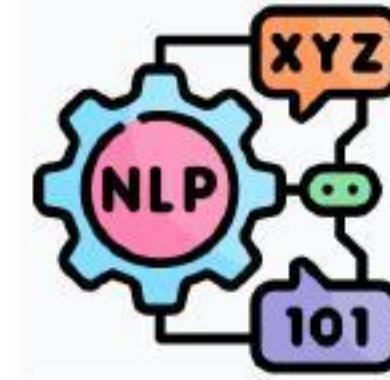
# Multi-Mask Language Model



# Multi-Mask Language Model



# Multi-Mask Language Model



## Loss function

Objective 1:  
Multi-Mask LM

Loss: Cross Entropy Loss



Objective 2:  
Next Sentence Prediction

Loss: Binary Loss

# MÔ HÌNH GOOGLE BERT



## I. Scaled-Dot Product & Multi-head Attention

## II. BERT

## III. BERT Objective

## IV. Subword Tokenization

## V. Finetuning BERT



# Subword Tokenization

Là một phương pháp chia nhỏ các từ thành các đơn vị nhỏ hơn gọi là subwords.



Thay vì chia từ như trong từ điển hay dùng dấu cách, subword tokenization tập trung vào việc tạo ra các subwords có ý nghĩa nhỏ hơn.



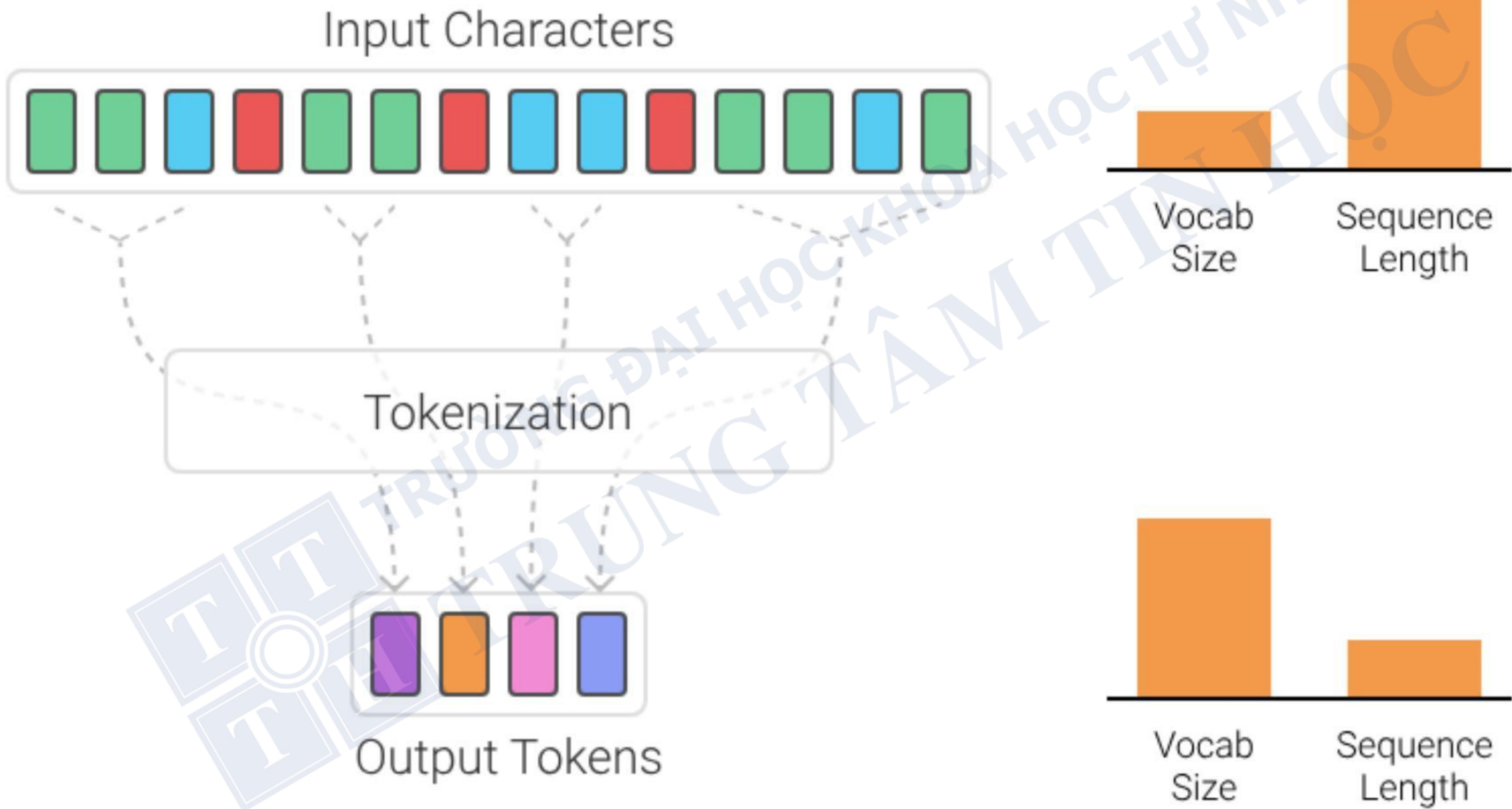
# Subword Tokenization

## Các bước thực hiện subword tokenization:

- 1. Xây dựng từ điển:** Từ điển được xây dựng từ tập dữ liệu train, bao gồm các từ và các subwords.
- 2. Phân tách từ thành subwords:** Các từ trong câu được chia thành các subwords dựa trên từ điển. Nếu một từ không tồn tại trong từ điển, nó sẽ được chia thành các subwords nhỏ hơn.
- 3. Gán mã cho subwords:** Mỗi subword được gán một mã số duy nhất để biểu diễn nó trong quá trình xử lý.



# Subword Tokenization





# Subword Tokenization

- Giả sử từ vựng của chúng ta gồm những từ sau:

vocabulary = [game, the, I, played, walked, enjoy]

- Xét câu input "I played the game". Vì tất cả các từ đều có trong từ vựng nên mã thông báo cuối cùng của chúng ta cho câu đã cho sẽ như sau:

tokens = [I, played, the, game]

- Xét một câu khác: "I enjoyed the game". Vì từ *enjoyed* không có trong từ vựng, chúng ta thay thế nó bằng một **unknown token <unk>**.

→ Do đó, mã thông báo cuối cùng là:

tokens = [ I, <UNK>, the, game ]



# Subword Tokenization

- Giả sử chúng ta chia từ **played** thành các từ phụ [play, ed] và từ **walked** được chia thành các từ phụ [walk, ed].

`vocabulary = [game, the, I, play, walk, ed, enjoy]`

- Xét câu: "I enjoyed the game". Kiểm tra thấy có các từ phụ **enjoy** và **ed** có trong từ vựng:

`tokens = [ I, enjoy, ##ed, the, game]`



# Subword Tokenization

## Mã hóa cặp byte(BPE):

Số lần xuất hiện của mỗi từ trong bảng là (cost, 2), (best, 2), (menu, 1), (men, 1), and (camel, 1).

Character sequence	Cost
C o s t	2
b e s t	2
m e n u	1
m e n	1
c a m e l	1



# Subword Tokenization

Giả sử chúng ta xây dựng từ điển có kích thước 14  
→ từ điển có 14 token.  
→ Tạo từ vựng bằng BPE.

Character sequence	Cost	Vocabulary
Co st	2	a, b, c, e, l, m, n, o, s, t, u
b e st	2	
me nu	1	
me n	1	
ca me l	1	

Character sequence	Cost	Vocabulary
Co st	2	a, b, c, e, l, m, n, o, s, t, u
b e st	2	
me nu	1	
me n	1	
ca me l	1	

Character sequence	Cost	Vocabulary
Co st	2	a, b, c, e, l, m, n, o, s, t, u, st
b e st	2	
me nu	1	
me n	1	
ca me l	1	

Character sequence	Cost	Vocabulary
Co st	2	a, b, c, e, l, m, n, o, s, t, u, st
b e st	2	
me nu	1	
me n	1	
ca me l	1	

Character sequence	Cost	Vocabulary
Co st	2	a, b, c, e, l, m, n, o, s, t, u, st, me
b e st	2	
me nu	1	
me n	1	
ca me l	1	

Character sequence	Cost	Vocabulary
Co st	2	a, b, c, e, l, m, n, o, s, t, u, st, me
b e st	2	
me nu	1	
me n	1	
ca me l	1	

Character sequence	Cost	Vocabulary
Co st	2	a, b, c, e, l, m, n, o, s, t, u, st, me, men
b e st	2	
me nu	1	
me n	1	
ca me l	1	



# Subword Tokenization

## WordPiece:

- Lấy ví dụ của BPE:

Character sequence	Cost	Vocabulary
C o s t	2	a, b, c, e, l, m, n, o,
b e s t	2	s, t, u
m e n u	1	
m e n	1	
c a m e l	1	

- Likelihood của cặp ký tự s và t là:

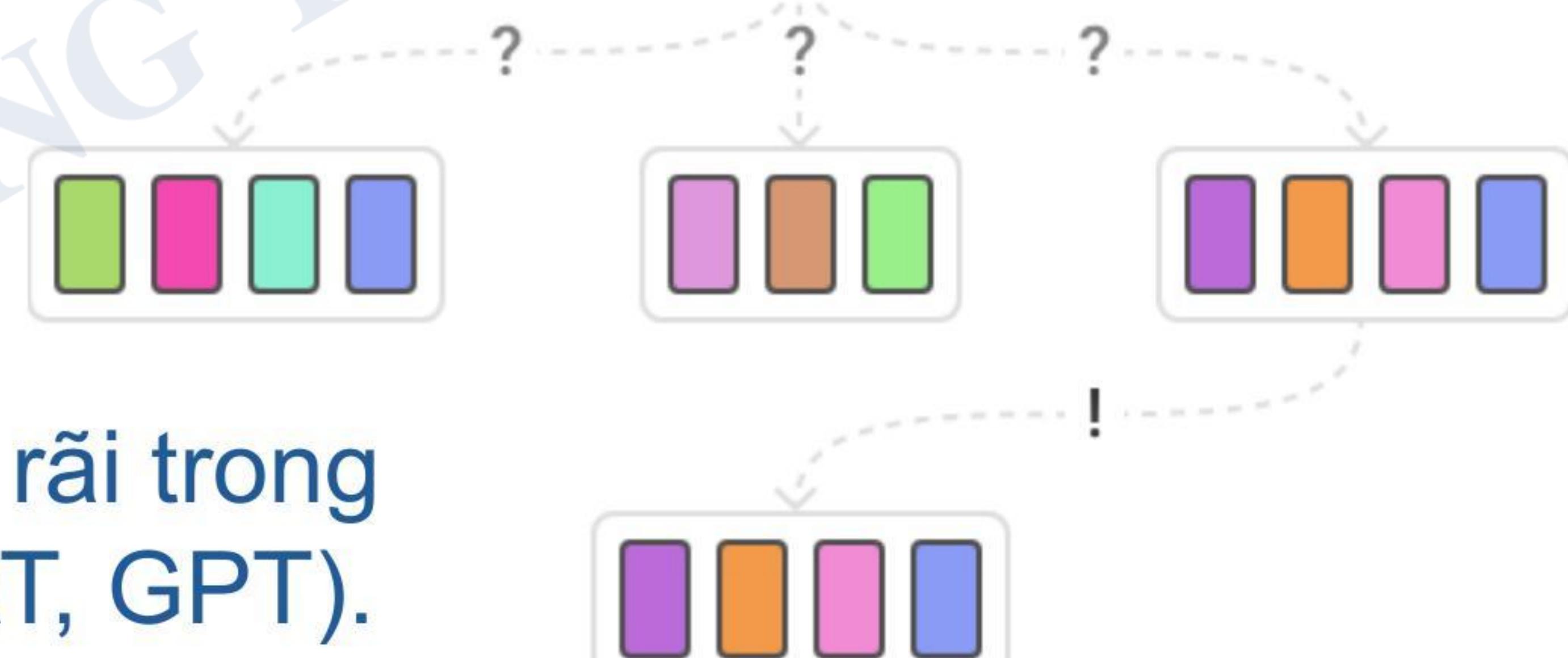
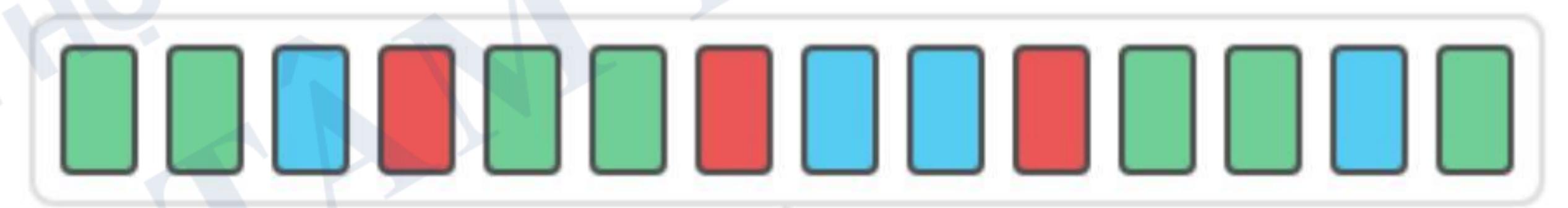
$$\frac{p(st)}{p(s)p(t)}$$



# Subword Tokenization

Giúp giảm kích thước từ vựng và xử lý các từ mới không có trong từ điển.

Giúp mô hình hiểu được các từ mới và các biến thể của từ thông qua việc biểu diễn chúng thành các subwords.



→ Được sử dụng rộng rãi trong các mô hình NLP (BERT, GPT).

# MÔ HÌNH GOOGLE BERT



## I. Scaled-Dot Product & Multi-head Attention

## II. BERT

## III. BERT Objective

## IV. Subword Tokenization

## V. Finetuning BERT



# Fine-tuning BERT

Là quá trình **điều chỉnh lại** mô hình BERT đã được train trước đó cho một tác vụ cụ thể hoặc tập dữ liệu mới.

→ Giúp mô hình BERT có thể học và áp dụng kiến thức đã học từ dữ liệu train ban đầu vào tác vụ cụ thể.

1. **Sentiment analysis**
2. **Natural language inference**
3. **Question - answering**



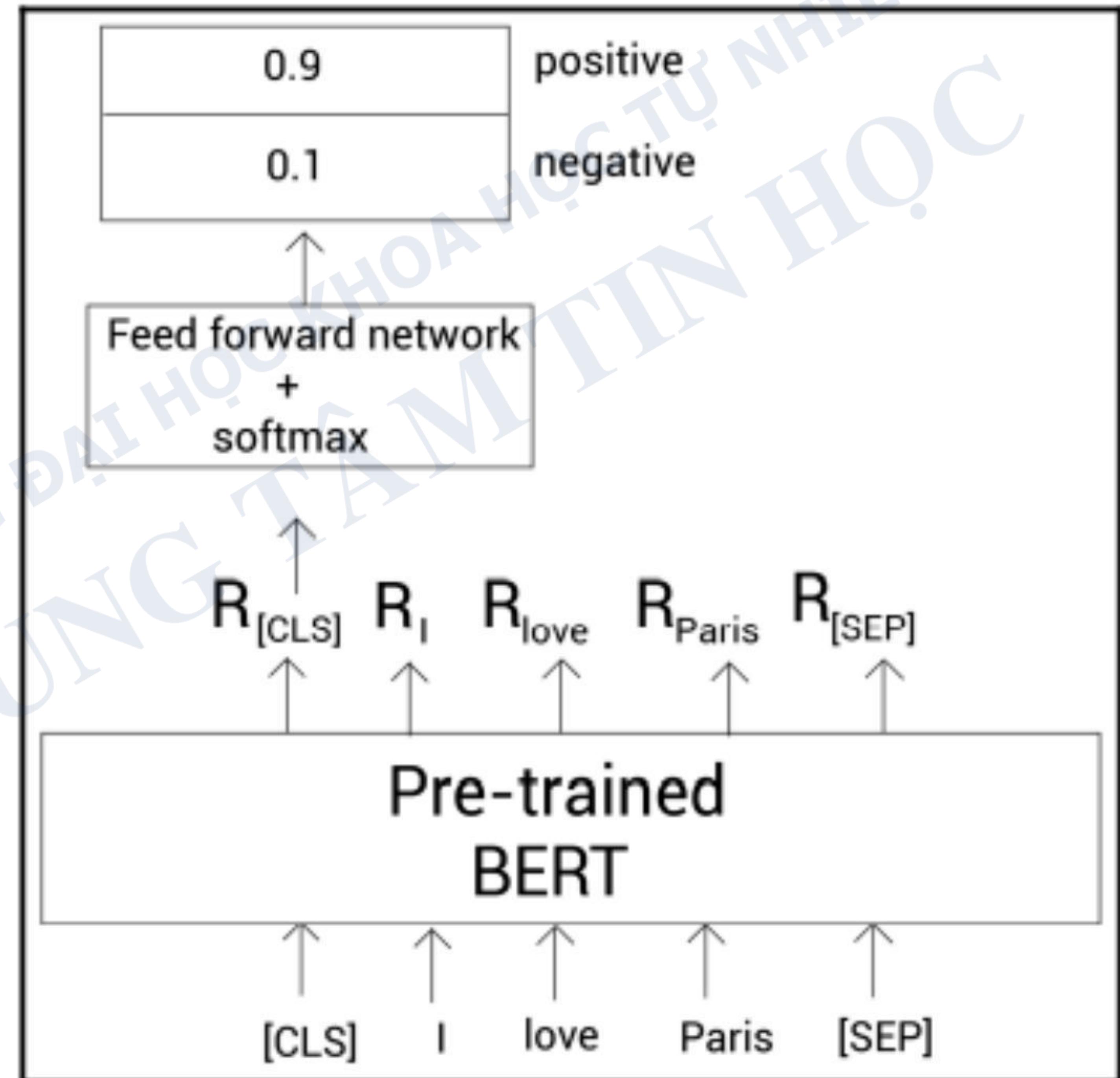
# Fine-tuning BERT

- 1. Khởi chạy mô hình:** Sử dụng mô hình BERT đã được huấn luyện trước đó, bao gồm các trọng số và kiến thức ngôn ngữ tự nhiên.
- 2. Thay đổi output layer** phù hợp với tác vụ cụ thể mà ta muốn fine-tune cho BERT.
- 3. Train với dữ liệu mới.**
- 4. Đánh giá và điều chỉnh:** đánh giá hiệu suất của mô hình. Điều chỉnh các tham số hoặc tiếp tục fine-tuning nếu cần thiết.

# Fine-tuning BERT



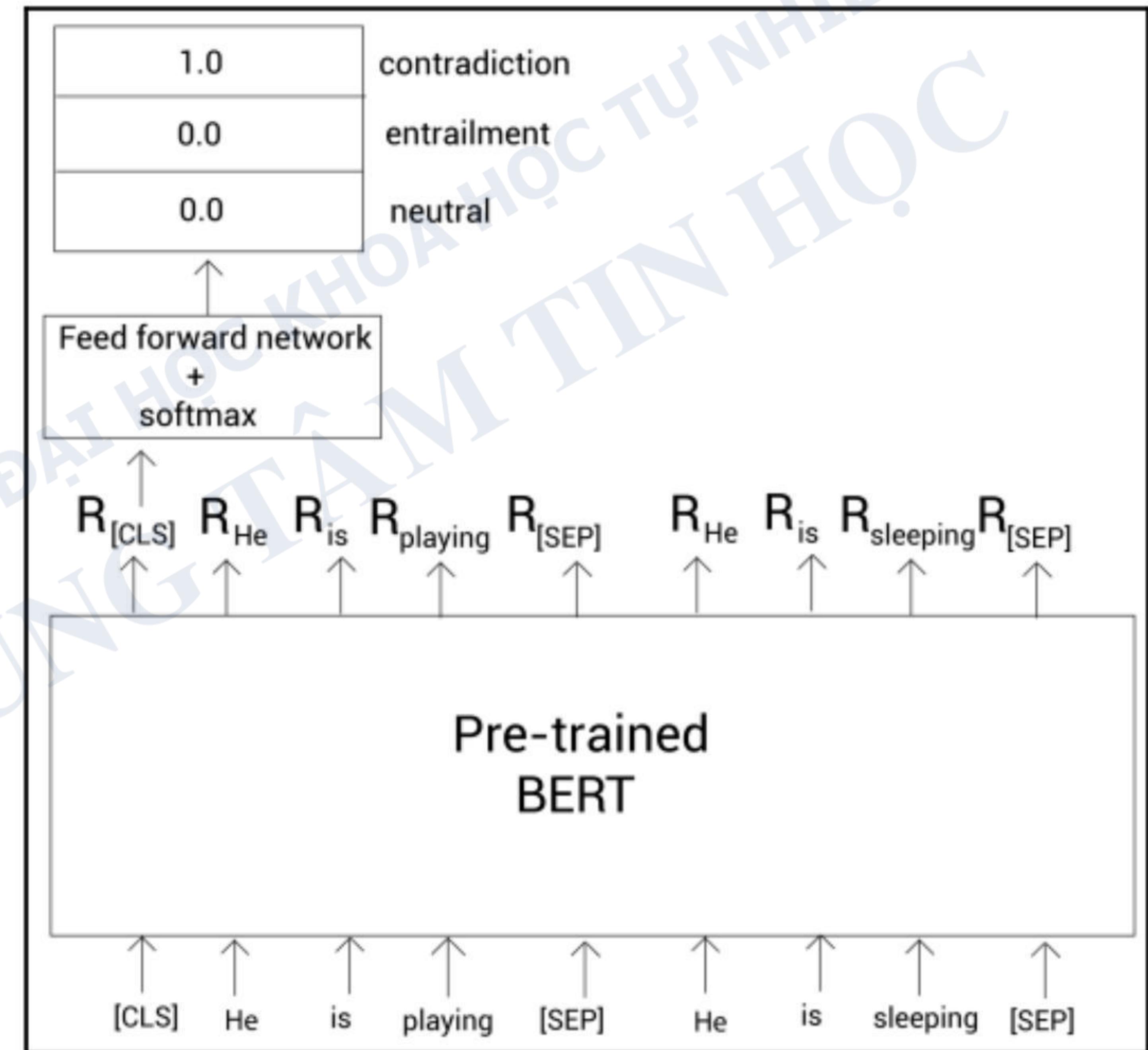
## Sentiment Analysis



# Fine-tuning BERT



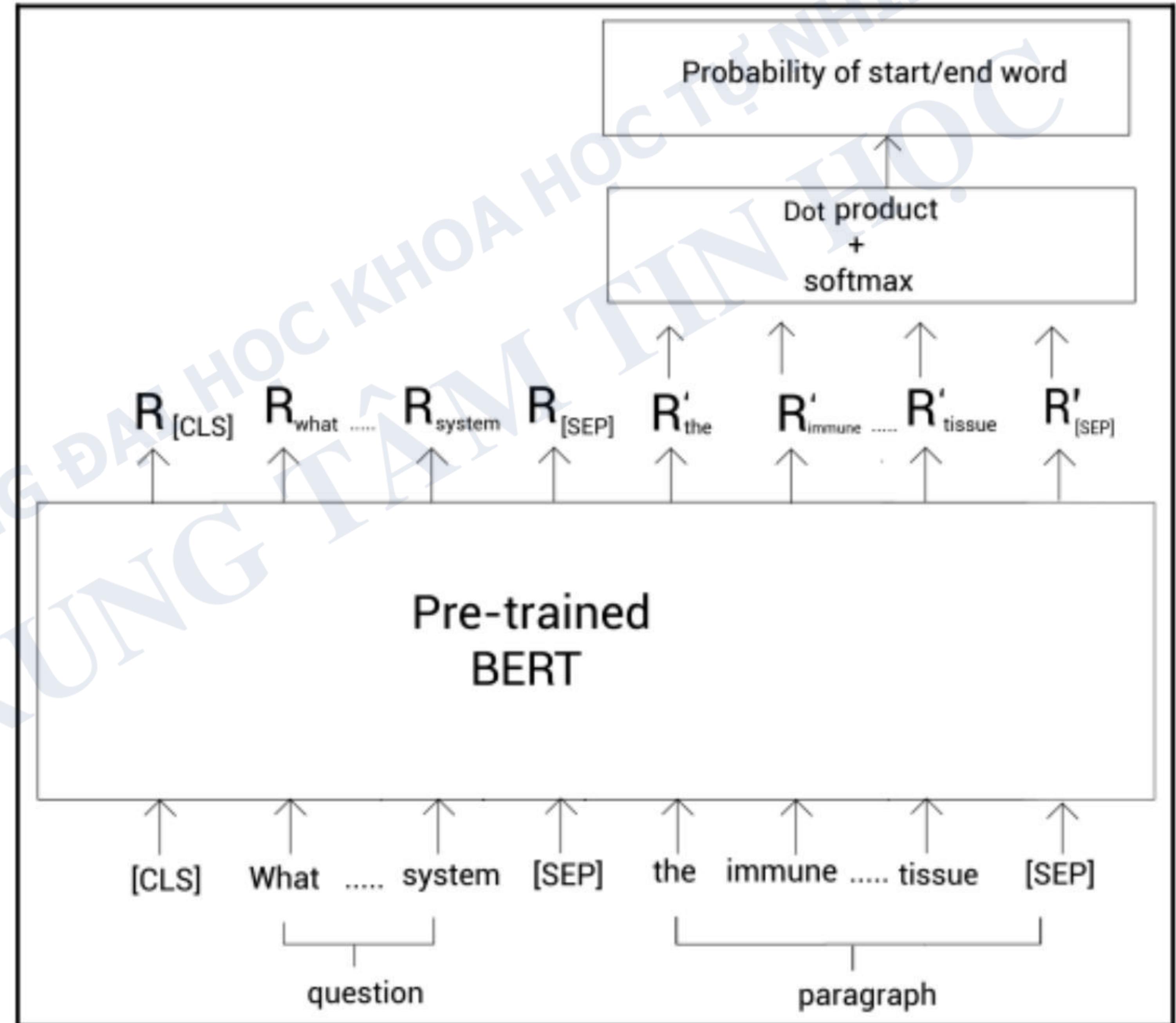
## Natural language inference



# Fine-tuning BERT



## Question- Answering





# Fine-tuning BERT

- Hãy xem xét cặp **question-answering** sau:

Question = "What is the immune system?"

Paragraph = "The immune system is a system of many biological structures and processes within an organism that protects against disease. To function properly, an immune system must detect a wide variety of agents, known as pathogens, from viruses to parasitic worms, and distinguish them from the organism's own healthy tissue."

- Mô hình cần trích xuất câu trả lời từ đoạn văn trên, và phải trả về đoạn văn bản có chứa câu trả lời:

Answer = "a system of many biological structures and processes within an organism that protects against disease"

Paragraph = "The immune system is a system of many biological structures and processes within an organism that protects against disease" biological structures and processes within an organism that protects against disease. To function properly, an immune system must detect a wide variety of agents, known as pathogens, from viruses to parasitic worms, and distinguish them from the organism's own healthy tissue."



# Fine-tuning BERT

## Giải thích Question-Answering

- Dùng hai vectors: start vector **S** và end vector **E**.
- Tính toán xác suất mỗi token (word) trong đoạn văn là **starting token** câu trả lời.

$$P_i = \frac{e^{S.R_i}}{\sum_j e^{S.R_j}}$$

- Tính toán **starting index** bằng cách chọn index của mã token có xác suất cao là starting token.

$$P_i = \frac{e^{E.R_i}}{\sum_j e^{E.R_j}}$$

# Code Demo



DEMO



# Q&A

---

