

# Bài 1: Khôi phục Masked Token (Masked Language Modeling)

```
from transformers import pipeline

# 1. Tạo pipeline "fill-mask"
# Pipeline này sẽ tự động tạo một mô hình mặc định phù hợp (thường là
# một biến thể của BERT)
mask_filler = pipeline("fill-mask")

# 2. Câu đầu vào với token <mask>
input_sentence = "Hanoi is the <mask> of Vietnam."

# 3. Thực hiện dự đoán
# top_k=5 yêu cầu mô hình trả về 5 dự đoán hàng đầu
predictions = mask_filler(input_sentence, top_k=5)

# 4. In kết quả
print(f"Câu gốc: {input_sentence}")

for pred in predictions:
    print(f"Dự đoán: '{pred['token_str']}' với độ tin cậy:
{pred['score']:.4f}")
    print(f" -> Câu hoàn chỉnh: {pred['sequence']}")
```

No model was supplied, defaulted to distilbert/distilroberta-base and revision fb53ab8 (<https://huggingface.co/distilbert/distilroberta-base>).

Using a pipeline without specifying a model name and revision in production is not recommended.

Some weights of the model checkpoint at distilbert/distilroberta-base were not used when initializing RobertaForMaskedLM:

- ['roberta.pooler.dense.bias', 'roberta.pooler.dense.weight']
  - This IS expected if you are initializing RobertaForMaskedLM from the checkpoint of a model trained on another task or with another architecture (e.g. initializing a BertForSequenceClassification model from a BertForPreTraining model).
  - This IS NOT expected if you are initializing RobertaForMaskedLM from the checkpoint of a model that you expect to be exactly identical (initializing a BertForSequenceClassification model from a BertForSequenceClassification model).

Device set to use cuda:0

Câu gốc: Hanoi is the <mask> of Vietnam.  
Dự đoán: ' capital' với độ tin cậy: 0.9341  
-> Câu hoàn chỉnh: Hanoi is the capital of Vietnam.  
Dự đoán: ' Republic' với độ tin cậy: 0.0300

```

-> Câu hoàn chỉnh: Hanoi is the Republic of Vietnam.
Dự đoán: ' Capital' với độ tin cậy: 0.0105
-> Câu hoàn chỉnh: Hanoi is the Capital of Vietnam.
Dự đoán: ' birthplace' với độ tin cậy: 0.0054
-> Câu hoàn chỉnh: Hanoi is the birthplace of Vietnam.
Dự đoán: ' heart' với độ tin cậy: 0.0014
-> Câu hoàn chỉnh: Hanoi is the heart of Vietnam.

```

Mô hình đã dự đoán đúng từ **capital** không: Mô hình có dự đoán đúng từ capital

**Tại sao các mô hình Encoder-only như BERT lại phù hợp cho tác vụ này** Các mô hình Encoder-only như BERT phù hợp cho các tác vụ này vì mô hình có thể học và hiểu ngữ cảnh từ cả 2 phía của từ cần dự đoán, giúp hiểu mối quan hệ của từ cần dự đoán với cả câu tốt hơn

## Bài 2: Dự đoán từ tiếp theo (Next Token Prediction)

```

from transformers import pipeline

# 1. Tạo pipeline "text-generation"
# Pipeline này sẽ tự động tạo một mô hình phù hợp (thường là GPT-2)
generator = pipeline("text-generation")

# 2. Đoạn văn ban đầu
prompt = "The best thing about learning NLP is"

# 3. Sinh văn bản
# max_length: tông độ dài của câu mới và phần được sinh ra
# num_return_sequences: số lượng chuỗi kết quả, mỗi nhận
generated_texts = generator(prompt, max_length=50,
num_return_sequences=1)

# 4. In kết quả
print(f"Câu mới: '{prompt}'")
for text in generated_texts:
    print("Văn bản được sinh ra:")
    print(text['generated_text'])

```

No model was supplied, defaulted to openai-community/gpt2 and revision 607a30d (<https://huggingface.co/openai-community/gpt2>).

Using a pipeline without specifying a model name and revision in production is not recommended.

```
{"model_id": "7ddbd60db833413a87648cabefc9a35", "version_major": 2, "version_minor": 0}
```

```
{"model_id": "58120bb7f6774b7ba37e05cb14f98e18", "version_major": 2, "version_minor": 0}
```

```
{"model_id": "f6d50a7d0b824682af127037acf4ab25", "version_major": 2, "version_minor": 0}

{"model_id": "d145c1c138d8466d9c7c8758999f574e", "version_major": 2, "version_minor": 0}

{"model_id": "9aad89d392b146bdbc7eb5909a720438", "version_major": 2, "version_minor": 0}

 {"model_id": "323e1f89caee484f893e42db0eb305bc", "version_major": 2, "version_minor": 0}

 {"model_id": "227d4eca6c5949e89f0f8a24aeb9860c", "version_major": 2, "version_minor": 0}
```

Device set to use cuda:0

Truncation was not explicitly activated but `max\_length` is provided a specific value, please use `truncation=True` to explicitly truncate examples to max length. Defaulting to 'longest\_first' truncation strategy. If you encode pairs of sequences (GLUE-style) with the tokenizer you can select this strategy more precisely by providing a specific strategy to `truncation`.

Setting `pad\_token\_id` to `eos\_token\_id`:50256 for open-end generation.

Both `max\_new\_tokens` (=256) and `max\_length` (=50) seem to have been set. `max\_new\_tokens` will take precedence. Please refer to the documentation for more information.

([https://huggingface.co/docs/transformers/main/en/main\\_classes/text\\_generation](https://huggingface.co/docs/transformers/main/en/main_classes/text_generation))

Câu nói: 'The best thing about learning NLP is'

Văn bản được sinh ra:

The best thing about learning NLP is the chance to get to know yourself. It will help you to understand the ideas behind your own work, and your own life.

I've taught a lot of people NLP. I've taught people how to learn, and how to understand.

It's not about which one of you will get to use your talents. It's how you'll develop yourself, and your own work. I'm not talking about how you'll learn a skill or technique, you'll learn a topic or concept, rather, you'll learn the whole world.

NLP is about learning and understanding.

NLP is about learning and understanding.

It's about learning and understanding and realizing that you're not going to know everything, you're going to learn.

```
I've taught people how to learn, and how to learn.  
I've taught people how to learn.
```

**Kết quả sinh ra:** hợp lý ở phần đầu, còn phần sau chưa hợp lý vì bị lặp câu khác nhiều, làm tổn token mà không đem lại nhiều ý nghĩa

**Mô hình Decoder-only như GPT lại phù hợp cho tác vụ này** Mô hình chỉ có ngữ cảnh của các từ phía trước để sinh ra từ tiếp theo, đảm bảo không bị lộ các từ sau từ trước khi training để tránh ảnh hưởng tới khả năng inference\*\*

## Bài 3: Tính toán Vector biểu diễn của câu (Sentence Representation)

```
import torch  
from transformers import AutoTokenizer, AutoModel  
  
# 1. Chọn một mô hình BERT  
model_name = "bert-base-uncased"  
tokenizer = AutoTokenizer.from_pretrained(model_name)  
model = AutoModel.from_pretrained(model_name)  
  
# 2. Câu đâ`u vào  
sentences = ["This is a sample sentence."  
  
# 3. Tokenize câu  
# padding=True: đệm các câu ngắn hơn để có cùng độ dài  
# truncation=True: cắt các câu dài hơn  
# return_tensors='pt': trả về kết quả dưới dạng PyTorch tensors  
inputs = tokenizer(sentences, padding=True, truncation=True,  
return_tensors='pt')  
  
# 4. Đưa qua mô hình để lấy hidden states  
# torch.no_grad() để không tính toán gradient, tiết kiệm bộ nhớ  
with torch.no_grad():  
    outputs = model(**inputs)  
  
# outputs.last_hidden_state chứa vector đầu ra của tất cả các token  
last_hidden_state = outputs.last_hidden_state
```

```

# shape: (batch_size, sequence_length, hidden_size)

# 5. Thực hiện Mean Pooling
# Đê' tính trung bình chính xác, chúng ta câ`n bo' qua các token đêm
# (padding tokens)
attention_mask = inputs['attention_mask']
mask_expanded = attention_mask.unsqueeze(-
1).expand(last_hidden_state.size()).float()
sum_embeddings = torch.sum(last_hidden_state * mask_expanded, 1)
sum_mask = torch.clamp(mask_expanded.sum(1), min=1e-9)
sentence_embedding = sum_embeddings / sum_mask

# 6. In kết quả
print("Vector biêu diễn của câu:")
print(sentence_embedding)
print("\nKích thước của vector:", sentence_embedding.shape)

{"model_id": "76159b9615f74b37b05cf41b1d1f40e", "version_major": 2, "version_minor": 0}

{"model_id": "4a01bbf704bd412383bf5e647e3f4a02", "version_major": 2, "version_minor": 0}

{"model_id": "cbddd19f54de471daf0e982f88989c7f", "version_major": 2, "version_minor": 0}

 {"model_id": "ecc0daf701bb4f0bb2cf4ce30b542fb", "version_major": 2, "version_minor": 0}

 {"model_id": "acd2e3be0c11413886971b94459964c6", "version_major": 2, "version_minor": 0}

Vector biêu diễn của câu:
tensor([-6.3874e-02, -4.2837e-01, -6.6779e-02, -3.8430e-01, -6.5784e-02,
        -2.1826e-01,  4.7636e-01,  4.8659e-01,  4.0647e-05, -7.4273e-02,
       -7.4740e-02, -4.7635e-01, -1.9773e-01,  2.4824e-01, -1.2162e-01,
       1.6678e-01,  2.1045e-01, -1.4576e-01,  1.2636e-01,  1.8635e-02,
       2.4640e-01,  5.7090e-01, -4.7014e-01,  1.3782e-01,  7.3650e-01,
      -3.3808e-01, -5.0331e-02, -1.6452e-01, -4.3517e-01, -1.2900e-01,
       1.6516e-01,  3.4004e-01, -1.4930e-01,  2.2422e-02, -1.0488e-01,
      -5.1916e-01,  3.2964e-01, -2.2162e-01, -3.4206e-01,  1.1993e-01,
      -7.0148e-01, -2.3126e-01,  1.1224e-01,  1.2550e-01, -2.5191e-01])

```

	-4.6374e-01, -2.7261e-02, -2.8415e-01, -9.9249e-02, -3.7017e-02,
	-8.9192e-01, 2.5005e-01, 1.5816e-01, 2.2701e-01, -2.8497e-01,
	4.5300e-01, 5.0945e-03, -7.9441e-01, -3.1008e-01, -1.7403e-01,
	4.3029e-01, 1.6816e-01, 1.0590e-01, -4.8987e-01, 3.1856e-01,
	3.2861e-01, -1.3403e-02, 1.8807e-01, -1.0905e+00, 2.1009e-01,
	-6.7579e-01, -5.7076e-01, 8.5947e-02, 1.9121e-01, -3.3818e-01,
	2.7744e-01, -4.0539e-01, 3.1305e-01, -4.1197e-01, -5.6820e-01,
	-3.9074e-01, 4.0747e-01, 9.9898e-02, 2.3719e-01, 1.0154e-01,
	-2.5670e-01, -2.0583e-01, 1.1762e-01, -5.1439e-01, 4.0979e-01,
	1.2149e-01, 1.9333e-02, -5.9029e-02, -2.0141e-01, 7.0860e-01,
	-6.4609e-02, 2.4779e-02, -9.0578e-03, 1.9666e-02, 3.0815e-01,
	-4.9832e-02, -1.0691e+00, 6.1072e-01, -4.9722e-02, -1.5156e-01,
	-6.7778e-02, 4.7812e-02, 5.2103e-01, 1.6951e-01, 1.0146e-02,
	5.3093e-01, -7.8189e-02, 6.5843e-02, -2.9382e-01, -4.6045e-01,
	4.2071e-01, 1.1822e-01, 2.3631e-01, -4.5379e-02, -1.3740e-01,
	-4.4018e-01, -6.8123e-02, 1.9935e-01, 8.7062e-01, -2.2603e-01,
	3.3604e-01, 2.0236e-01, 3.7898e-01, 1.9533e-01, -3.0366e-01,
	3.8633e-01, 6.1949e-01, 6.8663e-01, -1.8968e-01, -3.6815e-01,
	-1.6616e-01, -7.0827e-02, -3.4610e-01, -8.5326e-01, 4.6645e-02,
	2.8512e-01, 1.0890e-01, 2.5938e-01, -4.2975e-01, 4.3345e-01,
	2.0637e-01, -3.8656e-01, -3.8187e-02, 3.6925e-01, 3.0130e-01,
	4.0251e-01, 1.2887e-01, -3.7689e-01, -3.4447e-01, -4.2116e-01,
	-1.0252e-01, -8.9737e-02, 4.7384e-01, 8.1717e-02, 1.5885e-01,
	7.6674e-01, 3.4493e-01, 9.8538e-04, 4.8932e-02, 2.6132e-01,
	3.8329e-02, -2.0036e-01, 2.6654e-01, 9.3773e-02, -4.6779e-01,

02,	-4.0519e-01, -4.4310e-01, 6.1268e-01, -1.8950e-01, -3.8333e-01,
	2.0583e-01, 1.5379e-01, -1.4664e-01, 5.3847e-01, -3.9618e-01,
	-2.0599e+00, 6.7052e-01, 2.1112e-01, -4.7306e-01, 3.4865e-01,
	-2.9919e-01, 5.4614e-01, -5.3924e-01, -2.4877e-01, -2.9070e-02,
	-2.0319e-01, -7.3275e-02, -3.8147e-01, -5.4454e-01, 3.5049e-01,
	-1.1249e-01, -2.1471e-01, -3.8439e-01, -1.0760e-01, -8.8821e-02,
	2.5263e-01, 2.1448e-01, 5.5799e-02, -6.5411e-02, 9.9837e-02,
	3.3435e-01, 2.4018e-01, 2.9875e-02, -1.1191e-01, 5.4330e-01,
	-5.5214e-01, 1.1125e+00, 5.4141e-01, -7.4160e-02, 3.5337e-01,
	1.2313e-01, 3.4855e-02, -2.8568e-01, -1.2517e-01, -4.4332e-02,
	1.3323e-01, -2.4995e-01, -4.9833e-01, 4.1959e-01, -3.1580e-01,
	6.1942e-01, 3.1113e-01, 4.8846e-01, 6.1518e-01, -3.6326e-02,
	2.1294e-02, -3.5715e-01, 5.9126e-01, 1.5102e-01, -2.9641e-01,
	2.9441e-01, -1.4138e-01, 1.1662e-01, -3.6223e-01, -1.4621e-01,
	6.5254e-02, 3.9270e-01, 3.8543e-01, -2.3996e-01, -3.1482e-01,
	-4.6860e-01, -1.1920e-01, 8.6236e-02, -3.4596e-02, -3.6275e-01,
	-3.9838e-01, -3.6006e-01, -1.9672e-01, -2.7738e-01, -4.1097e-01,
	3.6456e-01, -2.6012e-01, 1.2587e-01, 1.2752e-01, 5.4261e-01,
	1.0569e-01, 3.5704e-01, 1.4766e-01, 4.4929e-01, -8.1255e-01,
	-3.0409e-02, 5.8063e-02, 2.0699e-01, 6.6129e-01, 3.9243e-01,
	-6.8644e-01, -8.3415e-01, -1.2653e-01, 1.9644e-01, -4.0900e-01,
	-6.3777e-02, -1.8780e-01, 7.9473e-02, -1.7443e-01, 3.1936e-01,
	3.6761e-01, 4.3044e-01, -1.7471e-01, 1.3718e-01, 1.4272e-01,
	-6.0642e-01, 2.3549e-01, 2.7794e-01, 1.0539e-01, -4.5836e-01,

	-3.2561e-01	1.5292e-02	-2.7672e-01	-4.8611e-01	3.9087e-01
	3.6016e-01	6.3403e-01	-1.2816e-01	-1.6720e-02	-3.0123e-01
	-1.7321e-01	-6.7296e-01	-2.7015e-01	-1.2534e-01	-8.0565e-01
	3.6115e-01	1.7370e-01	-3.5578e-01	-2.1725e+00	-2.8102e-02
	-2.6773e-02	-2.2444e-01	3.1249e-02	6.4420e-02	-1.5017e-01
	-3.4460e-01	-5.5676e-01	1.8039e-01	-4.2200e-01	-9.1074e-01
	-3.1339e-03	7.2439e-01	3.9006e-01	-4.4129e-02	-4.4785e-02
	2.8707e-02	-1.2432e-01	6.9166e-01	-1.3227e-02	-2.3540e-02
	-7.0615e-02	-4.5062e-01	4.5705e-01	3.3198e-01	-2.2727e-01
	3.2434e-01	-4.5709e-01	-5.1586e-01	-1.5693e-01	-1.0897e-01
	3.9317e-01	-2.5950e-01	-1.5326e-01	3.3276e-01	3.2522e-01
	-2.5241e-01	4.7946e-01	-3.7339e-01	-2.8146e-01	7.7628e-02
	2.7131e-01	-3.7212e-01	6.1400e-01	-2.9269e-01	-4.4389e-01
	-3.7750e-01	2.7135e-01	3.6869e-01	-1.6904e-01	-1.7583e-01
	2.9626e-01	2.9393e-01	-8.2036e-03	3.4545e-02	4.5846e-01
	3.0137e-01	1.6171e-01	-2.7772e-01	5.2397e-01	-6.1950e-01
	-2.4818e-02	-5.1942e-02	3.6764e-01	-5.8404e-01	-2.6651e-01
	-7.5761e-02	-1.7428e-01	4.1535e-01	-2.7556e-01	-5.6796e-02
	-4.3509e-01	-9.6659e-01	-1.1800e-01	-3.8004e-01	2.7555e-01
	-2.9743e-01	2.4023e-01	-3.8869e-01	-4.0248e-01	-8.3882e-01
	-1.0652e-01	-9.4192e-02	1.4810e-01	9.0844e-03	1.4658e-01
	-1.4813e-01	-1.6078e-01	-4.3130e-01	-8.0683e-02	4.3722e-01
	4.2623e-01	3.3201e-01	-2.8283e-01	2.0751e-01	5.9093e-01
	-6.3453e-01	5.7386e-01	-2.9870e-01	1.0221e-02	-4.7624e-01
	4.9509e-01	4.7470e-02	1.3193e-01	3.6281e-01	-

1.1642e+00,					
	3.8372e-01,	1.7071e-01,	3.8881e-01,	1.7703e-01,	-4.7019e-01,
	1.2768e-01,	-1.3409e-01,	-2.8794e-01,	3.2066e-01,	-3.7853e-01,
	4.6259e-01,	5.2343e-01,	3.0741e-01,	2.7410e-01,	4.9933e-01,
	-5.6466e-01,	-3.4677e-01,	-6.6571e-01,	-1.3347e-01,	-8.5910e-02,
	6.2487e-02,	-3.9922e-01,	-3.5880e-01,	-5.8337e-01,	-1.3556e-02,
	-1.6812e-01,	1.3949e-01,	2.9142e-01,	-4.5623e-01,	-1.0705e-01,
	6.6569e-01,	7.6614e-01,	-1.9306e-01,	4.3854e-01,	2.8110e-01,
	-3.6835e-01,	-1.6012e-01,	-2.5005e-01,	7.6297e-01,	1.9653e-01,
	-1.8120e-01,	1.1895e-03,	1.8755e-01,	-1.8990e-01,	-2.3725e-01,
	3.2633e-02,	-2.7723e-01,	-4.7986e-02,	-6.2332e-01,	2.6807e-01,
	-1.2293e-01,	-2.7098e-01,	-6.9677e-01,	1.5738e-01,	5.3557e-01,
	1.2760e-01,	-1.7979e-02,	1.2769e-01,	-5.6453e-02,	6.7965e-02,
	1.8555e-01,	-3.6374e-01,	2.8518e-01,	-4.3920e-01,	-2.4276e-01,
	5.1755e-01,	-2.3519e-01,	6.4010e-02,	3.9268e-01,	5.7986e-01,
	-1.7500e-01,	7.1669e-02,	5.7915e-01,	5.1699e-02,	-1.1085e-03,
	-4.8444e-02,	1.5531e-01,	2.8402e-01,	6.8268e-01,	8.1524e-02,
	1.5325e-01,	1.9466e-01,	1.2260e-02,	-3.3223e-01,	2.5763e-02,
	-1.6071e-01,	-3.7663e-01,	-7.3670e-01,	-5.0067e-01,	1.1540e-01,
	-3.3788e-01,	1.2889e-01,	2.1528e-02,	6.1149e-01,	3.3550e-01,
	-2.0217e-01,	-6.3961e-02,	2.4056e-02,	-9.3070e-02,	-2.7771e-02,
	1.8373e-01,	-4.1812e-02,	-1.0456e-01,	-2.7569e-01,	-3.9216e-01,
	-3.2092e-01,	-1.0158e+00,	1.6407e-01,	4.5044e-02,	2.3079e-01,
	2.6936e-02,	-2.1047e-01,	-3.1392e-01,	-4.6154e-01,	-4.0347e-01,
	7.3271e-02,	1.1470e-01,	-2.4129e-01,	-3.6199e-01,	-5.3254e-01,

	-5.2185e-01, -4.0713e-01, 2.1619e-02, 1.4186e-01, -1.2105e-01,
	-1.4055e-02, -4.2986e-02, -1.2459e-01, -6.6652e-01, -6.4169e-01,
	-2.2399e-01, 6.2557e-02, -3.3323e-01, 1.8865e-02, 1.6465e-01,
	-2.8729e-02, -5.9477e-01, 2.0963e-02, -3.3761e-01, 1.8088e-01,
	7.4363e-01, 1.5554e-01, 2.7824e-01, -2.1975e-01, 5.1316e-01,
	-3.9708e-01, -2.4769e-01, 4.3027e-01, -2.3078e-01, -2.9392e-01,
	1.3250e-01, -6.1646e-01, 2.6501e-01, 5.6891e-01, -1.3585e-01,
	-1.2774e-01, 8.1189e-01, 3.6497e-01, 5.0178e-01, 2.9736e-01,
	8.7772e-01, 7.3390e-02, 2.5788e-01, -3.3609e-01, 8.8207e-02,
	2.1282e-02, 1.4487e-01, 7.6676e-03, -3.9123e-01, -6.3919e-02,
	-3.7236e-01, 8.2942e-02, 3.0821e-02, 3.1530e-02, 2.0262e-01,
	-5.0065e-01, -1.2373e-01, 2.2661e-01, 1.6069e-01, -3.6415e-01,
	2.3418e-01, -1.6900e-01, -1.3540e-01, -1.6677e-01, 1.5227e-01,
	-2.6064e-01, 4.4845e-02, -3.4592e-02, -1.2043e-01, 6.4724e-01,
	4.8944e-01, -3.0347e-01, -2.3118e-01, -8.3765e-02, 2.2163e-01,
	1.0404e-01, 1.3495e-01, -5.3097e-01, 1.4525e-01, 4.9890e-01,
	-4.9265e-01, 3.7358e-01, 2.2077e-01, -5.4249e-02, -6.7141e-02,
	6.2194e-01, 4.6524e-01, -4.2303e-01, -3.2715e-01, 3.8370e-01,
	-5.7111e-01, -1.6922e-01, 4.2353e-01, -2.0156e-01, -1.2482e-01,
	4.3334e-01, -4.0269e-02, -5.8663e-01, 7.2658e-01, -5.5645e-01,
	-5.7467e-02, -2.1052e-01, 1.0038e-01, -2.5418e-03, 7.7563e-01,
	-3.9355e-01, 6.4184e-01, -5.9658e-01, 2.1974e-02, 1.8323e-01,
	1.7593e-01, 4.8541e-01, -4.6240e-01, 3.5692e-01, 3.2622e-01,
	-2.0756e-01, 5.7904e-01, -2.7194e-01, -5.2925e-01, 7.4888e-02,
	-2.6069e-02, 3.5997e-01, 5.5750e-01, 3.2160e-01, 4.0078e-

```

01,      5.1017e-01, -4.6595e-02,  2.9056e-01,  2.4928e-01,  2.0993e-
01,
        4.9611e-01, -4.1696e-02, -1.5711e-01,  1.5638e-01,  8.1300e-
02,
        3.2564e-01, -2.6684e-01, -2.1355e-01,  1.9676e-01,  4.6960e-
01,
        1.5972e-01, -2.5918e-01, -1.0547e-01,  1.3562e-01,  3.5989e-
01,
       -1.0882e-01, -7.1567e-02, -5.3039e-01,  8.8760e-01, -3.4283e-
01,
       -5.0051e-02, -4.8836e-01,  2.0944e-01,  2.6859e-01,  4.4360e-
01,
       -4.6622e-01, -1.3640e-01, -1.4363e-01, -3.5663e-01, -1.1210e-
01,
       -1.9890e-01, -1.2909e-01, -3.0789e-03, -6.2015e-02, -4.2345e-
01,
        2.7059e-01, -3.1317e-01,  5.7516e-01, -2.2513e-03,  1.7034e-
01,
        3.9410e-01,  8.1126e-01, -3.6260e-01,  5.2088e-01, -5.4591e-
01,
       -5.8637e-02,  1.5576e-01,  1.7441e-01,  1.3422e-01, -4.4368e-
01,
        2.6824e-01, -2.6424e-01, -5.6734e-01,  2.7222e-01,  5.5829e-
01,
       -9.1910e-01,  2.2039e-01, -3.5612e-01,  1.3164e-01, -1.1517e-
01,
       -2.0684e-01, -2.7871e-02,  3.9112e-01, -6.6897e-01, -3.8353e-
01,
       -5.6089e-02,  8.0477e-01, -2.5700e-01, -1.0725e-01,  7.5041e-
02,
        2.4736e-01, -6.1457e-01, -1.9508e-01,  5.4606e-01,  3.3887e-
01,
        2.7338e-01,  4.4597e-01,  4.4805e-01, -7.3450e-01,  2.2959e-
01,
       -3.8097e-02, -1.4963e-01, -2.4957e-01, -2.8457e-01,  5.6483e-
01,
       5.4733e-02,  8.0649e-02, -1.2184e+00,  5.7510e-01,  1.3625e-
01,
       -4.4055e-01,  6.9751e-02, -4.0260e-01,  1.0932e-01, -6.6830e-
02,
       -3.9555e-02, -5.4193e-01, -4.4191e-01,  2.4927e-01,  6.6517e-
01,
       -1.7534e-01, -1.2388e-01,  3.1970e-01]])

```

Kích thước của vector: torch.Size([1, 768])

- **Kích thước (chiều) của vector biểu diễn là: 768**
- **Con số này tương ứng với tham số của mô hình BERT là: số chiều của tầng ẩn**

- **Cần sử dụng attention\_mask khi thực hiện Mean Pooling vì: bỏ qua các token đệm (padding tokens)**