**BERT**

# **1.Tổng quan về BERT**

## **1.1. BERT là gì?**

BERT, viết tắt của **Bidirectional Encoder Representations from Transformers**, là một mô hình Machine Learning (Học máy) dành cho xử lý ngôn ngữ tự nhiên (NLP). Nó được phát triển vào năm 2018 bởi các nhà nghiên cứu tại Google AI Language và được xem như một “con dao đa năng” cho hơn 11+ tác vụ ngôn ngữ phổ biến, chẳng hạn như phân tích cảm xúc ,nhận dạng thực thể có tên, trả lời câu hỏi, tóm tắt, giải quyết từ đa nghĩa, …

## **1.2. BERT sinh ra để làm gì?**

Các phương pháp biểu diễn từ truyền thống như TF-IDF, Word2Vec, GloVe có hạn chế lớn là Vector tĩnh (static embeddings): Mỗi từ chỉ có một vector duy nhất, không thay đổi theo ngữ cảnh → không xử lý được đa nghĩa (polysemy).  
 → Ví dụ: “đi” trong *“đi du lịch”* ≠ “đi nhanh”.

Vì vậy, ta cần biểu diễn động (contextual embeddings) – vector thay đổi theo ngữ cảnh, và biểu diễn sâu (deep representations) để hiểu ngôn ngữ phức tạp hơn.

**Giai đoạn chuyển tiếp – Mô hình dựa trên LSTM**

* ELMo (2018): Dùng LSTM hai chiều, học từ dữ liệu không nhãn → vector từ thay đổi theo ngữ cảnh.  
   Cải thiện đáng kể (5–10%) nhưng chậm, khó song song hóa.
* ULMFit: Dùng pre-training + fine-tuning trên LSTM, áp dụng kỹ thuật *gradual unfreezing* để tránh overfitting.

Hạn chế: vẫn dựa trên RNN, gặp vấn đề *vanishing gradient*, không tận dụng GPU hiệu quả.

**Cách mạng Transformer (2018)**

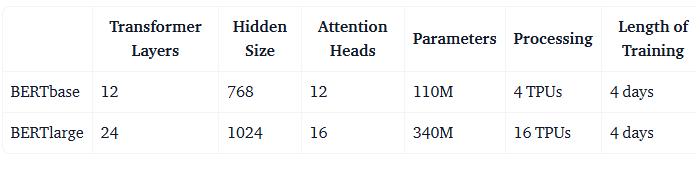
Ra đời từ bài báo “Attention is All You Need”, thay thế hoàn toàn RNN bằng Self-Attention. Hai mô hình phổ biến:

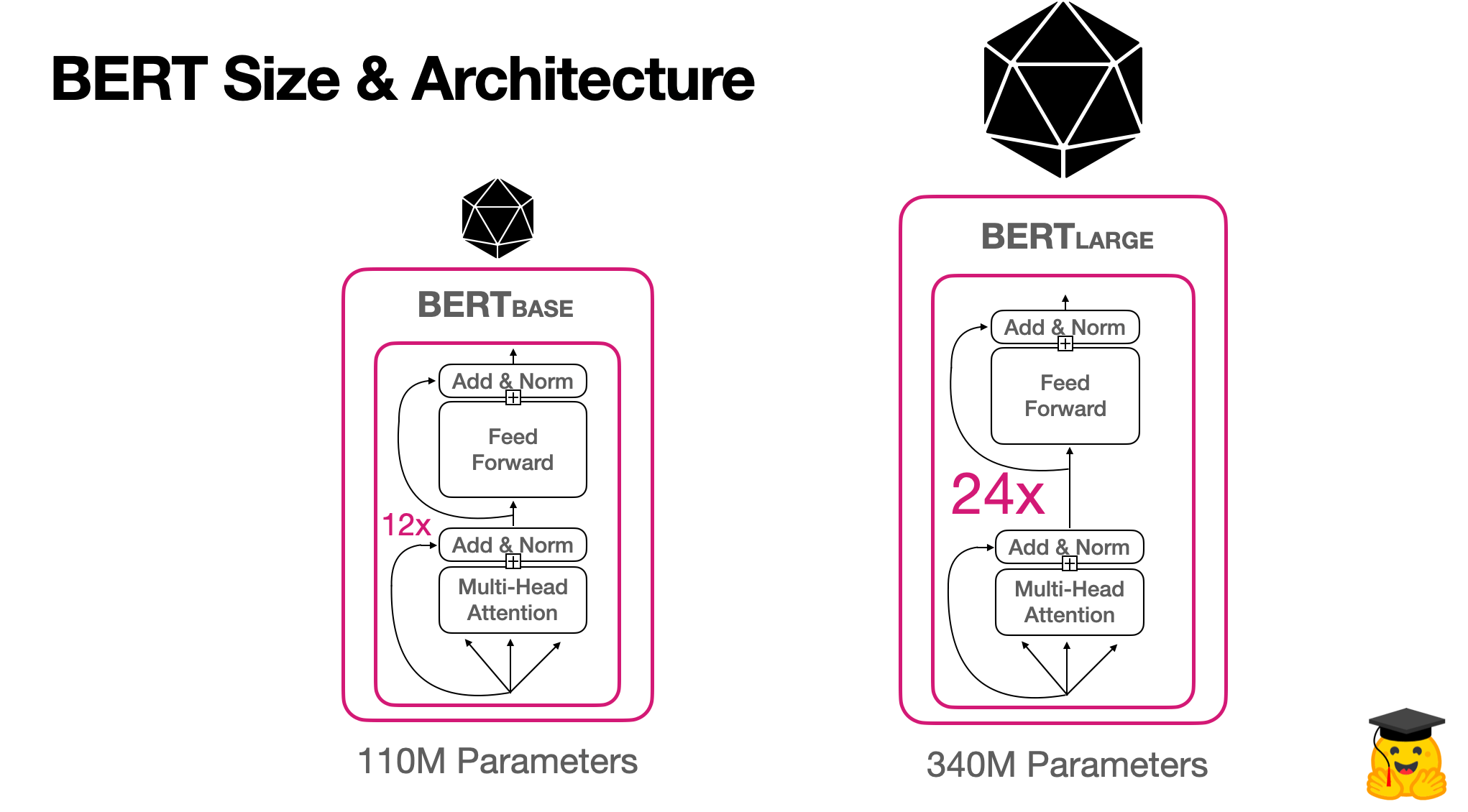
**-Bert**-Sử dụng Encoder

**-GPT**-Sử dụng Decoder

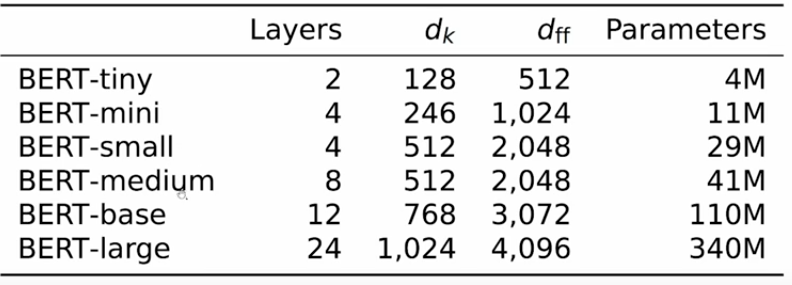
# **2. Kiến trúc của Bert**

Thông số kỹ thuật của hai mô hình BERT gốc:



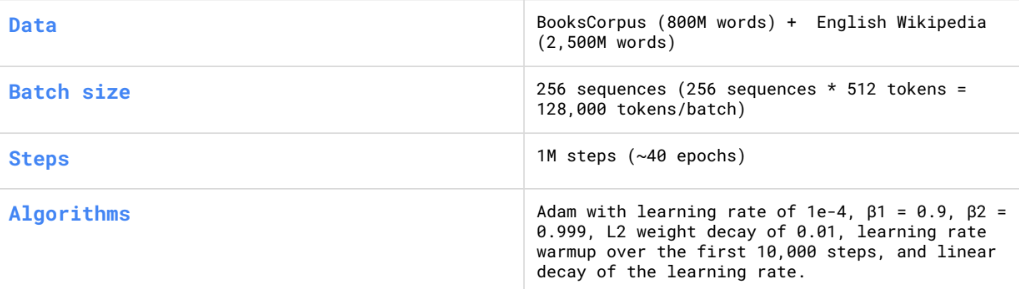


Các mô hình Bert khác:



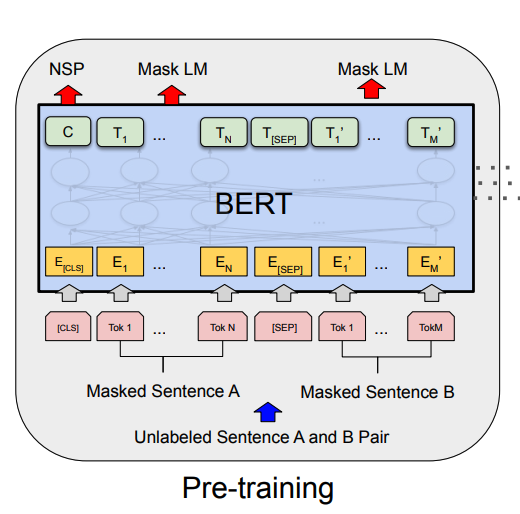
# **3. Pre-train Bert**

BERT được pre-train trên **3,3 tỷ từ** đã góp phần tạo nên thành công vượt trội của BERT.



Dữ liệu **văn bản thô** (Wikipedia, BooksCorpus, Common Crawl, v.v.) có **vô hạn**,  
 nhưng dữ liệu **có nhãn thủ công** (cho QA, classification, NER, …) thì rất ít.

Vì vậy Bert sử dụng 2 cách học Self-supervised learning là MLM (Masked Language Modeling) và NSP (Next Sentence Prediction) giúp **học biểu diễn ngôn ngữ mạnh mẽ** từ dữ liệu khổng lồ mà **không cần label**.



## **3.1: Masked Language Model**

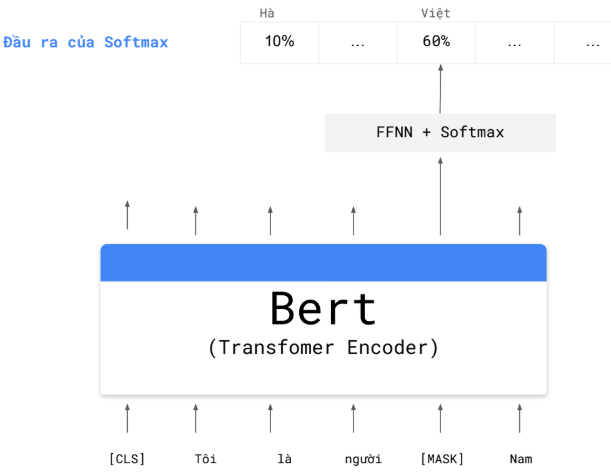
Model được train từ trái qua phải và từ phải qua trái. Điều này cho hiệu năng cao hơn model chỉ train từ trái qua phải như mô hình ngôn ngữ GPT.

Để train mô hình này, tác giả đã ngẫu nhiên che đi (mask) một lượng % tokens trong văn bản sau đó huấn luyện lượng tokens còn lại để dự đoán ra tokens bị che này.

Cụ thể tác giả đã che đi 15% số lượng WordPiece Tokens ở mỗi chuỗi đầu vào một cách ngẫu nhiên. Trong 15% này, tác giả:

* Thay thế 80% bằng token [MASK]
* Thay thế 10% bằng token ngẫu nhiên
* Giữ nguyên 10% còn lại

Tác giả lý giải cách làm này là do token [MASK] chỉ xuất hiện khi train pre-trained và không xuất hiện khi chúng ta fine-tune.



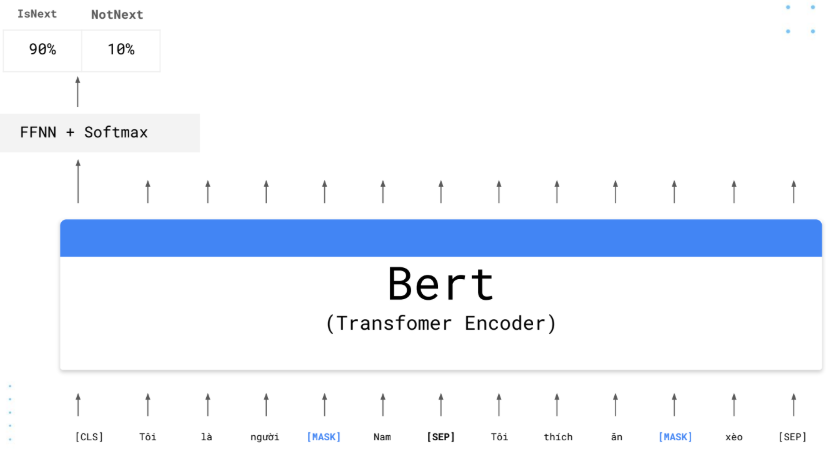
## **3.2. Next Sentence Prediction (NSP)**

**NSP** giúp BERT học về **mối quan hệ giữa các câu** bằng cách **dự đoán xem một câu có thực sự nối tiếp câu trước đó hay không.**

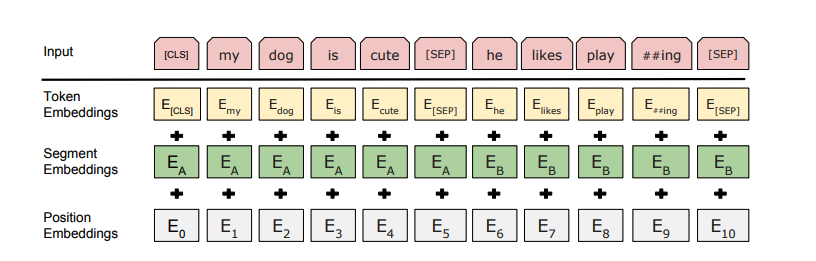
Dữ liệu được train sẽ bao gồm các cặp câu A-B trong đó:

* 50% câu B đi theo sau câu A với nhãn dự đoán là IsNext
* 50% câu B ngẫu nhiên từ trong ngữ liệu với nhãn dự đoán là NotNext

Giúp BERT **học cách phân biệt sự liên kết giữa các câu.**

****

## **3.3. Cách BERT tạo embedding đầu vào cho mô hình**



| Trong đó: |  |
| --- | --- |
| **Token Embeddings** | Vector biểu diễn nội dung từng token (từ hoặc subword) |
| **Segment Embeddings** | Phân biệt **câu A** và **câu B** trong cặp câu (phục vụ nhiệm vụ NSP) |
| **Position Embeddings** | Mã hóa **vị trí thứ tự của token trong chuỗi** |

# **3. Đánh giá mô hình**

Trong paper gốc, sau khi BERT được **pre-train** xong (MLM + NSP), nhóm tác giả phải **đánh giá xem BERT hiểu ngôn ngữ tốt đến mức nào**.  
 Để làm điều này, họ:

* Fine-tune BERT trên **một loạt benchmark NLP chuẩn quốc tế**
* So sánh với **các mô hình tốt nhất lúc đó (SOTA – State-of-the-Art)**.

**Quy trình cụ thể:**

1.Fine-tune trên mỗi task riêng

* Mỗi task có một **lớp đầu ra riêng** (output head).
* BERT-base và BERT-large đều được fine-tune **end-to-end** (tất cả tham số được cập nhật).
* Learning rate nhỏ (2e-5 đến 5e-5), batch size 16–32, 3–4 epochs.

2. Sử dụng metric chuyên biệt cho từng task

Ví dụ:

* **Classification task** → Accuracy hoặc F1
* **Semantic similarity** → Pearson / Spearman correlation
* **QA (SQuAD)** → F1 và Exact Match (EM)

3. So sánh với baseline

So sánh với:

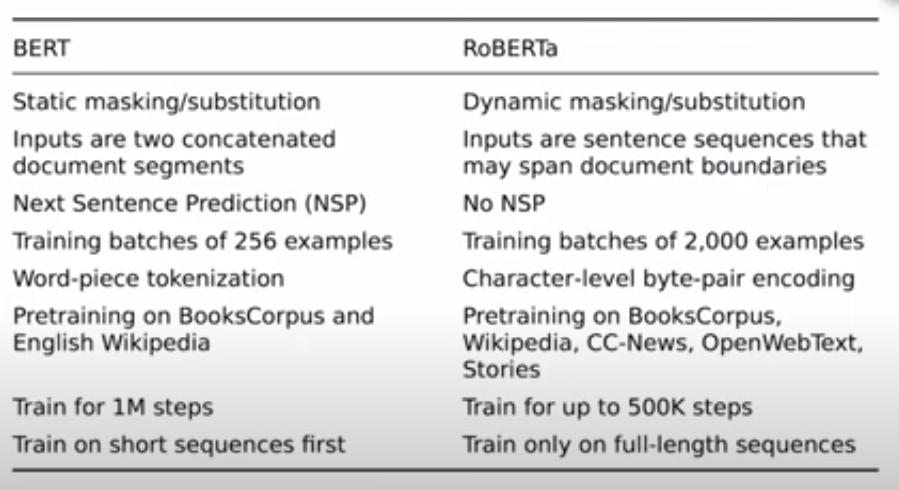
* GPT (Generative Pre-trained Transformer)
* ELMo
* OpenAI Transformer
* Các mô hình RNN, CNN khác

BERT **vượt tất cả** với biên độ rõ rệt, đặc biệt ở các task yêu cầu **hiểu ngữ cảnh 2 chiều (bi-directional context)**.

# **4. Một số biến thể của BERT**

## **4.1.RoBERTa**

**Mục tiêu:** Tối ưu lại quy trình huấn luyện của BERT, không thay đổi kiến trúc.



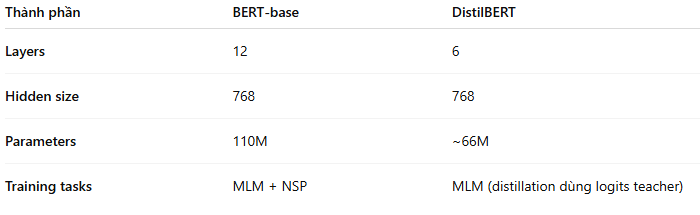
**Kết quả:** RoBERTa outperform BERT trên hầu hết các benchmark như **GLUE, SQuAD, RACE**.

## **4.2. DistilBERT (Distilled BERT)**

**Mục tiêu:** Làm mô hình **nhẹ hơn** mà vẫn giữ được **đa phần sức mạnh của BERT**.

**Kỹ thuật: Knowledge Distillation**

* Dùng BERT gốc (gọi là *teacher model*) để huấn luyện một mô hình nhỏ hơn (*student model*).
* Student học bằng cách:  
  + Bắt chước **phân phối xác suất (logits)** của teacher.
  + Học **embedding representation** tương tự.
  + Giữ lại **MLM objective**.



**Hiệu quả:**

* Nhanh hơn 60%,
* Nhẹ hơn 40%,
* Giữ ~97% hiệu năng của BERT-base.