BERT
DEBERTA
DEBERTA-V3

Bộ dữ liệu gốc

Bộ dữ liệu bao gồm 1.378 bài luận, một số được viết bởi học sinh và một số được tạo sinh bằng các LLM.
Cấu trúc dữ liệu train:

train_essay.csv

- *id*: id của từng sample.
- *prompt_id*: chỉ ra prompt mà đoạn văn bản được dùng để trả lời.
- *text*: đoạn văn bản.
- *generated*: đoạn văn bản được viết bởi học sinh (0) hay LLM (1), đây là giá trị cần dự đoán.

train_prompt.csv

- *prompt_id*: id của prompt.
- *prompt_name*: tựa đề của prompt.
- *instructions*: hướng dẫn cho học sinh.
- source_text: văn bản của đoạn prompt.

Bộ dữ liệu bổ trợ

BỘ DỮ LIỆU BỔ TRỢ

Bộ dữ liệu bổ trợ DAIGT | External Dataset bao gồm 2421 bài luận được viết bởi học sinh và 2421 bài luận được sinh ra bởi ChatGPT.
Cấu trúc bộ dữ liệu:

- *id*: id của từng sample.
- *text*: đoạn văn bản trích xuất từ cuộc thi Feedback Prize 3, được xem như bài luận được viết bởi học sinh.
- *instructions*: đoạn prompt hướng dẫn cho ChatGPT.
- *source_text*: bài luận được ChatGPT tạo sinh.

Bộ dữ liệu bổ trợ

BỘ DỮ LIỆU BỔ TRỢ

Bộ dữ liệu bổ trợ DAIGT | External Dataset bao gồm 2421 bài luận được viết bởi học sinh và 2421 bài luận được sinh ra bởi ChatGPT.
Cấu trúc bộ dữ liệu:

- *id*: id của từng sample.
- *text*: đoạn văn bản trích xuất từ cuộc thi Feedback Prize 3, được xem như bài luận được viết bởi học sinh.
- *instructions*: đoạn prompt hướng dẫn cho ChatGPT.
- *source_text*: bài luận được ChatGPT tạo sinh.

Bộ dữ liệu bổ trợ

Do bộ dữ liệu bổ trợ chứa 2 class đồng đều nhau nên chúng ta sẽ so sánh dựa trên bộ dữ liệu bổ trợ này:

- Các đoạn văn do người viết thường có những lỗi ngữ pháp, cách diễn đạt nội dung chưa thật sự tốt, có một số đoạn văn nhiều lỗi lặp từ do chủ yếu do học sinh từ lớp 6 đến lớp 12 viết.
- Ngược lại các đoạn văn bản do ChatGPT viết sẽ bay bổng hơn do được huấn luyện trên rất nhiều dữ liệu.

=> Giúp cho mô hình có thể đạt được accuracy cao.

Bộ dữ liệu bổ trợ

Ví dụ: do học sinh viết: Online classes, i think that if students really trying to learn something is not need for them to take classes online but sometimes, teachers cannot teach al the students well because is only 1 teacher and like 30 students in every class room, and i think that online classes are not a bad idea because they been teach by a computer and they have to pay attention or they will fail, but at school they wont fail that easy because they can get copy from another students and thats how they will pass classes better at school then online .

Now lets really talk about online classes, is a perfect solution for students that for some reason they cant go to school in case that they had a car accident they cant walk but they parents dont want them to fail the school year , or they dont feel like going to school thats why online classes are the best choice not only for me for a lot of people like student and parents; I mean school is not that bad at school students can make more and more friends every year online they will get tired if they have to see the same computer and hear the same voice everyday.

Bộ dữ liệu bổ trợ

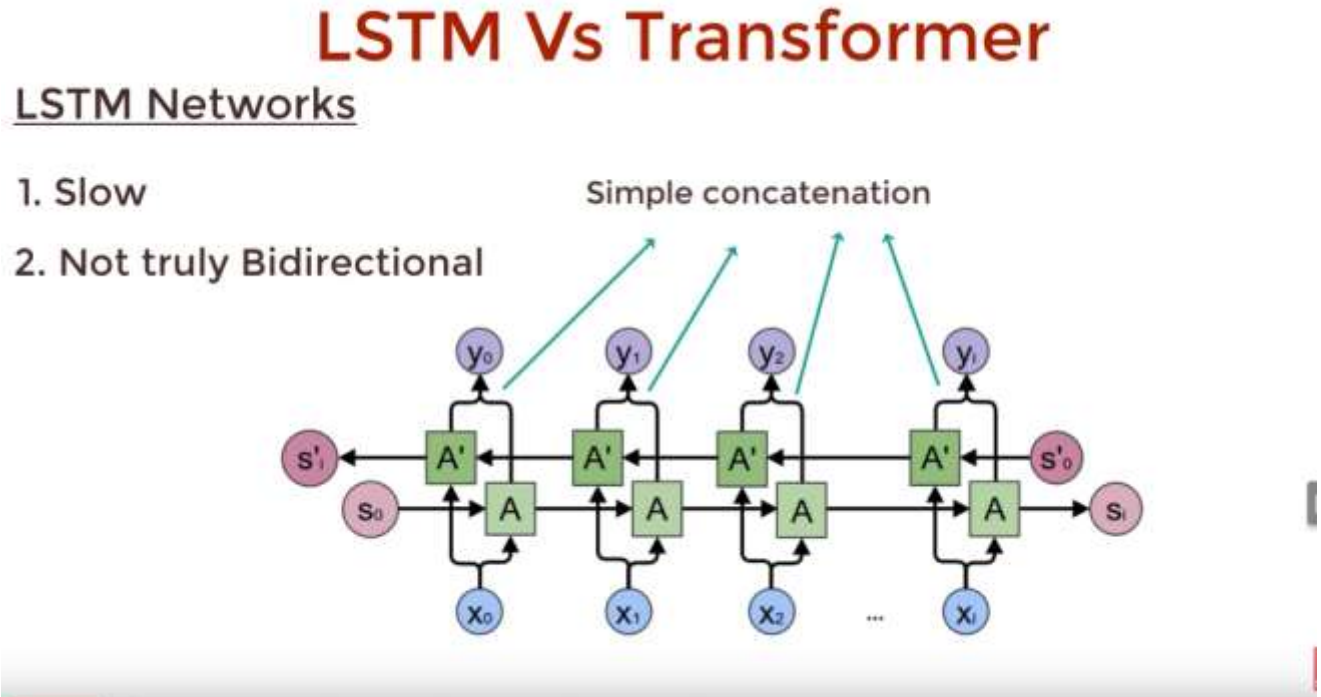
Ví dụ: do ChatGPT viết: Online classes have been increasingly popular as technology advances and the need for social distancing increases. There are several advantages and disadvantages to taking courses online. One pro is that online classes can be taken at any time and from any location, making them more accessible to people who have difficulty attending a standard class. There is also evidence that online classes can foster student engagement as students are in control of when and how they access their education. Additionally, online classes make it easier for students to collaborate and communicate with one another in discussion forums or chat rooms.

On the other hand, there are several drawbacks to taking online classes. It can be difficult for students to motivate themselves when there is no professor present and no regular deadlines and expectations. Online classes also often lack the personal connections and relationships that come from face-to-face instruction. Furthermore, several studies have shown that students tend to be less successful in passing online classes than in traditional classroom courses.

In conclusion, while there are some advantages to taking online classes, there are also disadvantages to consider. Ultimately, the decision of whether or not to take online classes should be made on an individual basis, taking into account student engagement, difficulty of passing, and student-teacher relationships."

BERT

Các mô hình hồi tiếp truyền thống không thể thật sự hiểu được ngữ cảnh thông tin 2 chiều và chậm để huấn luyện do phụ thuộc vào các timestep trước đó.



BERT

Mục đích

- Giúp hiểu ngữ cảnh xung quanh mỗi từ trong câu, không chỉ dựa vào từ đó đơn lẻ.
- Đưa toàn bộ câu vào 1 lần.
- Biểu diễn ngôn ngữ của BERT giúp nâng cao khả năng xác định ý nghĩa của từ và cụm từ.
- Classification Task, Question Answering, Named Entity Recognition…

BERT

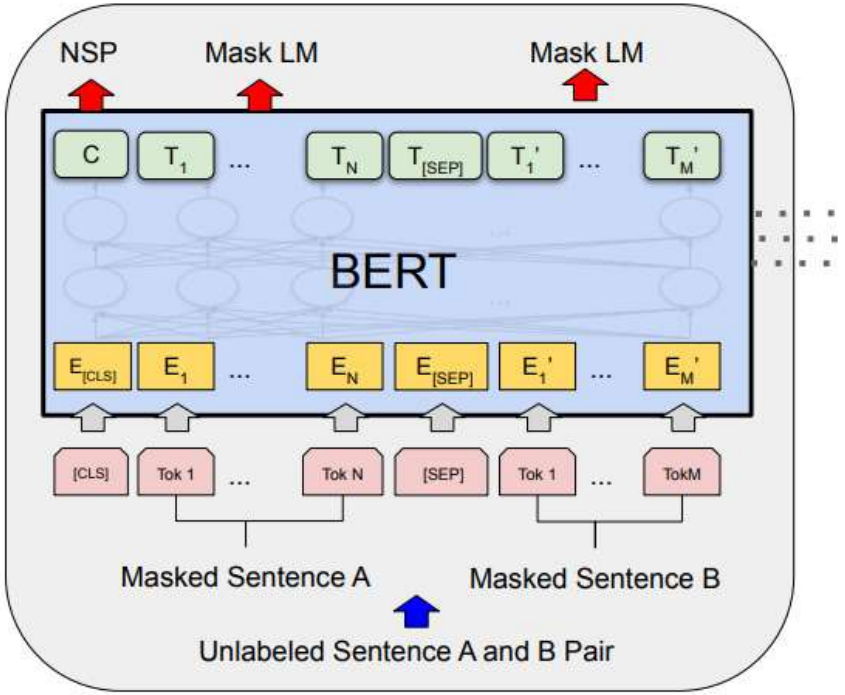
Hướng tiếp cận

- Pre-training cho biểu diễn hai chiều thì tốt hơn so với biểu diễn một chiều (từ trái sang phải / phải sang trái) hay kết hợp hợp cả biểu diễn trái sang phải và phải sang trái.
- Implementation gồm 2 bước **pre-training** (train trên unlabeled data) để học các biểu diễn ngữ cảnh của từ và **fine-tuning** (dùng labeled data cho các task cụ thể)

BERT

Cách hoạt động (quá trình pre-training)

- Chuỗi tokens được đưa vào khối Transformer encoder. Những chuỗi này đầu tiên sẽ được embedding, và sau đó được đưa qua mạng neural. Output sẽ gồm chuỗi các vector embedding với mỗi vector (chứa thông tin ngữ cảnh) ứng với 1 token trong chuỗi đầu vào.



Pre-training

BERT

Cách hoạt động (quá trình pre-training)

- Chuỗi tokens được đưa vào khối Transformer encoder. Những chuỗi này đầu tiên sẽ được embedding, và sau đó được đưa qua mạng neural. Output sẽ gồm chuỗi các vector embedding với mỗi vector (chứa thông tin ngữ cảnh) ứng với 1 token trong chuỗi đầu vào.

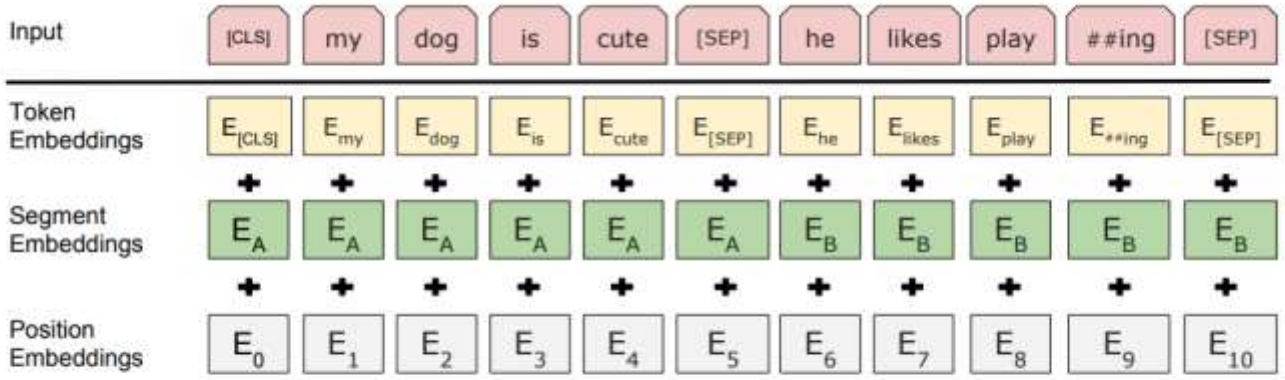


Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

BERT

Cách hoạt động (quá trình pre-training)

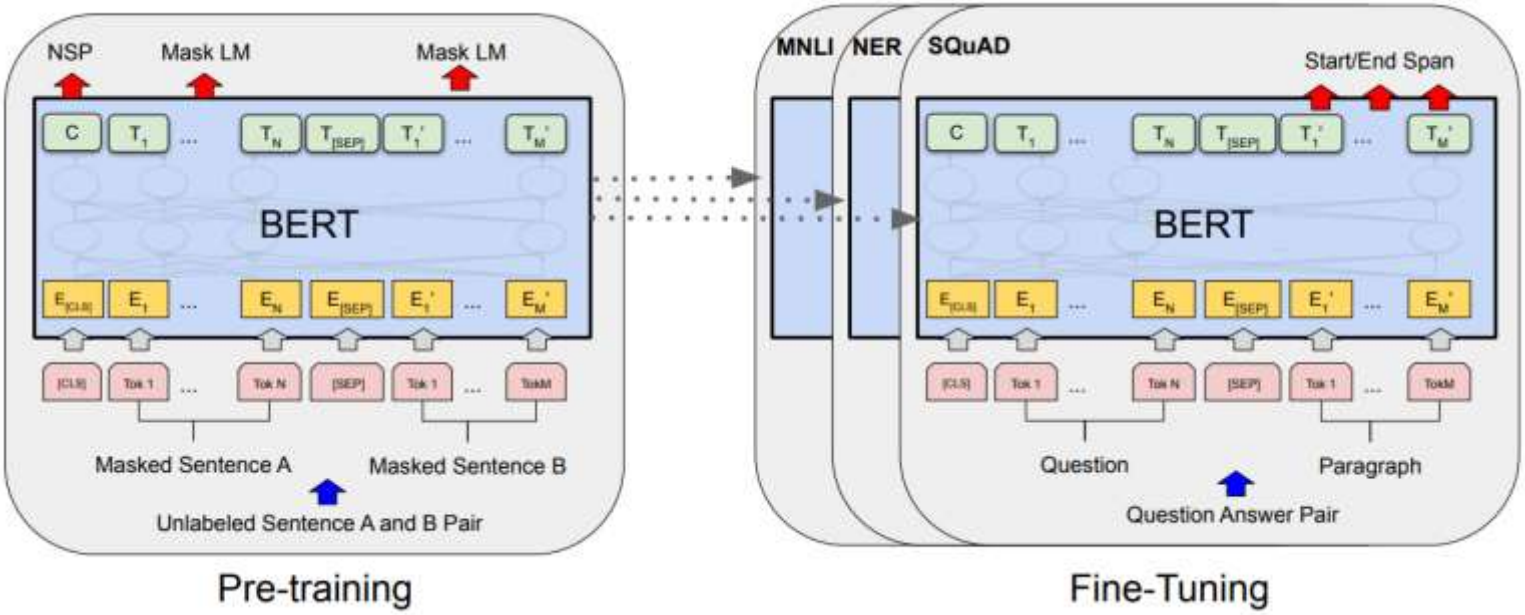
- Nhiều mô hình dự đoán từ tiếp theo trong 1 chuỗi 1 cách tuần tự (cách làm này có thể hạn chế việc học ngữ cảnh của mô hình). Bert giải quyết vấn đề này thông qua 2 chiến lược training:

(1) Masked Language Model (MLM) dùng để học mối tương quan của các phụ thuộc gần, giữa các từ trong câu với nhau.
(2) Next Sentence Prediction (NSP) dùng để học mối tương quan của các phụ thuộc xa, giữa các câu khác nhau.

BERT

Fine-tuning

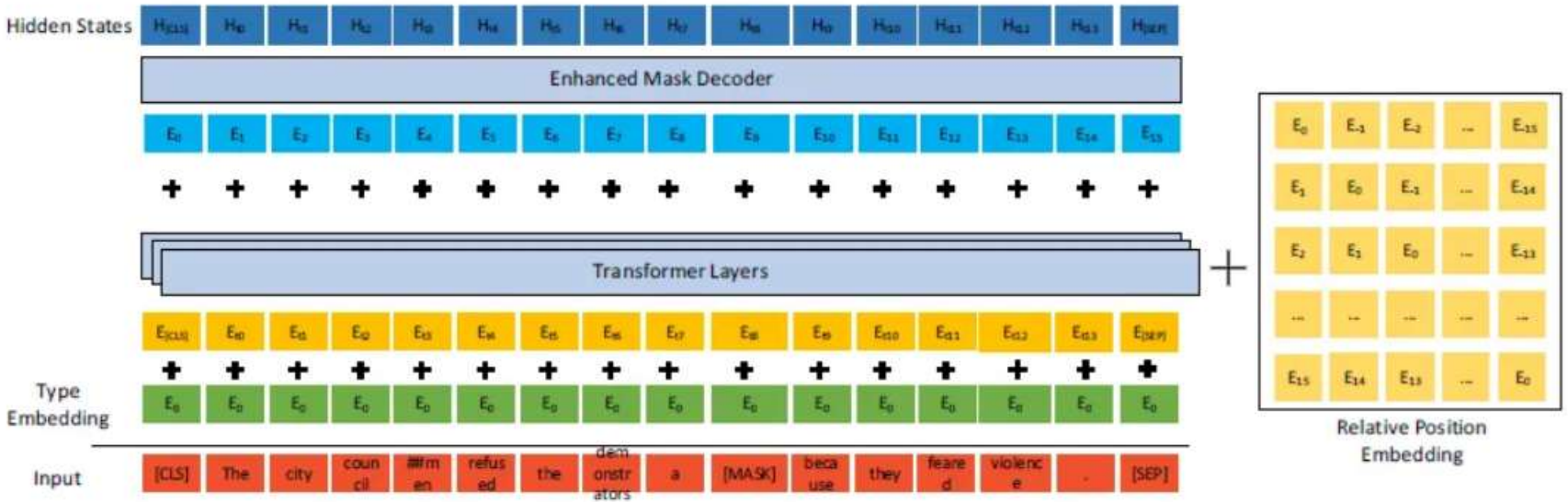
- Sau khi được pretrain, mô hình sẽ được fine-tuning cho các task cụ thể



DEBERTA

Cải tiến của BERT với 2 cơ chế mới:

- Disentangled attention
- Incorporate absolute positions in the decoding layer



The model architecture of DeBERTa (Figure from authors' slides)

DEBERTA

Disentangled attention

- Với mỗi token ở vị trí **i** trong chuỗi, nó sẽ được biểu diễn thông qua 2 vector, ***{Hi}*** biểu diễn nội dung và ***{Pi\|j}*** biểu diễn vị trí tương đối với token ở vị trí **j**
- Cross attention score giữa tokens i và j có thể được tính theo công thức sau:

$$
\begin{aligned}
A_{i,j} &= \{H_i, P_{i|j}\} \times \{H_j, P_{j|i}\}^\intercal \\
&= H_i H_j^\intercal + H_i P_{j|i}^\intercal + P_{i|j} H_j^\intercal + P_{i|j} P_{j|i}^\intercal
\end{aligned}
$$

DEBERTA

Disentangled attention

- Trong mỗi head attention bình thường, các phép toán chỉ gồm có:

$$Q = HW_q, K = HW_k, V = HW_v, A = \frac{QK^\intercal}{\sqrt{d}}$$

$$H_o = \text{softmax}(A)V$$

- Trong khi đó, các phép tính trong **Disentangled Attention** bao gồm:

$$Q_c = HW_{q,c}, K_c = HW_{k,c}, V_c = HW_{v,c}, Q_r = PW_{q,r}, K_r = PW_{k,r}$$

$$\tilde{A}_{i,j} = \underbrace{Q_i^c K_j^{c\intercal}}_{\text{(a) content-to-content}} + \underbrace{Q_i^c K_{\delta(i,j)}^{r\intercal}}_{\text{(b) content-to-position}} + \underbrace{K_j^c Q_{\delta(j,i)}^{r\intercal}}_{\text{(c) position-to-content}}$$

$$H_o = \text{softmax}(\frac{\tilde{A}}{\sqrt{3d}})V_c$$

DEBERTA

Incorporate absolute positions in the decoding layer

- EMD gồm 2 input là **I** và **H.**
- **n** là số lớp EMD. Ở lớp đầu tiên (n=1), **I** mang thông tin vị trí tuyệt đối, **H** là output của lớp Transformer trước đó. Ở lớp thứ hai (n=2), **I** là output của lớp EDM đầu tiên.
- Chỉ sử dụng trong quá trình pretraining cho task Masked Language Model.

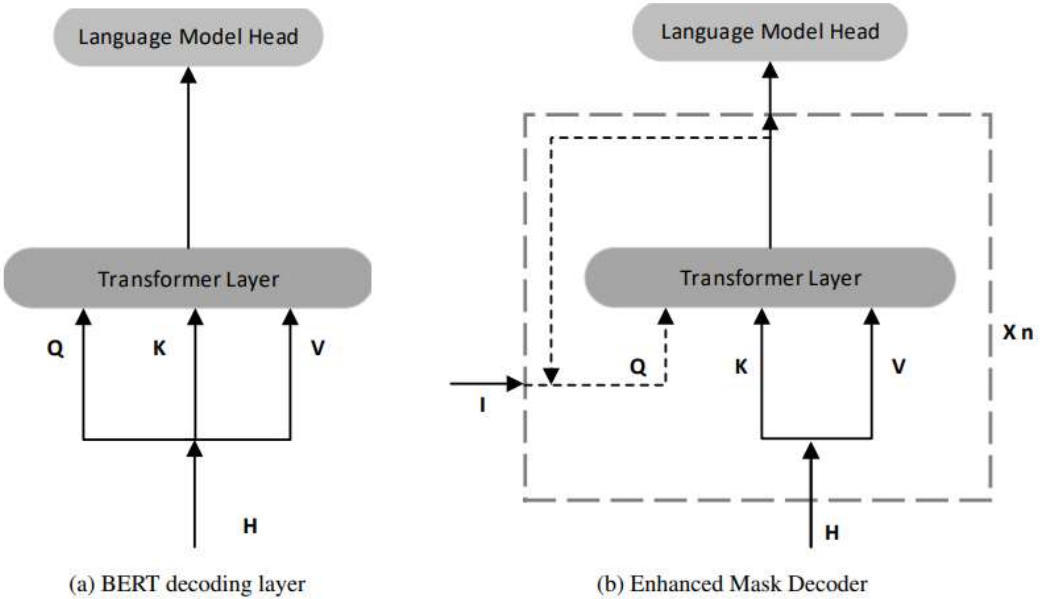
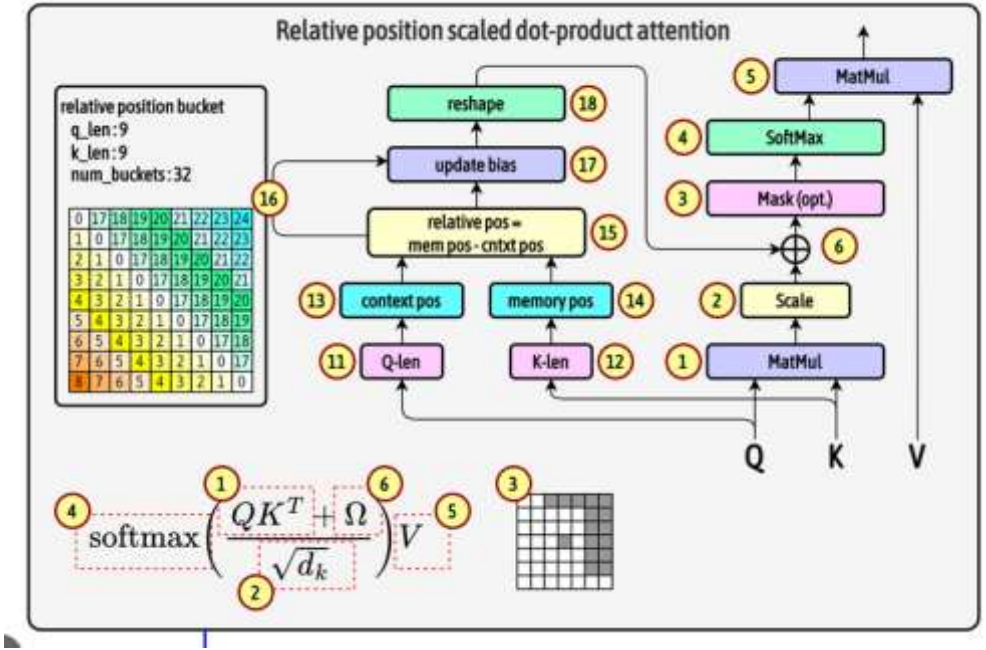


Figure 2: Comparison of the decoding layer.

DEBERTA-V2

- Sử dụng **sentencepiece tokenzier** với 128k vocabulary.
- Thêm 1 lớp conv trước lớp attention để học tốt hơn phụ thuộc cục bộ.
- Chia sẻ trọng số của ma trận vị trí và ma trận nội dung.
- Sử dụng bucket để encode thông tin vị trí tương đối tương tự như T5.



SentencePiece Tokenizer

- SentencePiece là một bộ tokenizer và detokenzier đơn giản, hiệu quả và không phụ thuộc vào ngôn ngữ, cung cấp cho bộ tokenizer khả năng tokenize không mất mát thông tin…
- Gồm 4 phần: Normalizer, Trainer, Encoder, and Decoder.
- Thay ký tự khoảng trắng bằng _ và xem nó như 1 ký tự, giúp cho có thể tokenize những ngôn ngữ không khoảng trăng như tiếng Nhật, từ đó cung cấp khả năng tokenize ko phụ thuộc ngôn ngữ.
- Sử dụng Byte Pair Ecndong và Unigram.
- Chuẩn hoá theo NKFC unicode.
- Hạn chế tối đa các unknown token.

Byte Pair Encoding

Phương pháp mới BPE (SOTA): Nhược điểm của phương pháp tokenize theo character level đó là các token không có ý nghĩa nếu đứng độc lập. Do đó đối với các bài toán sentiment analysis, áp dụng tokenize theo character level sẽ mang lại kết quả kém hơn. Token theo word level cũng tồn tại hạn chế đó là không giải quyết được các trường hợp từ ngầm ngoài từ điển.

Một phương pháp mới đã được đề xuất trong bài báo Neural Machine Translation of Rare Words with Subword Units vào năm 2016, có khả năng tách từ theo level nhỏ hơn từ và lớn hơn ký tự được gọi là subword. Phương pháp đó chính là BPE (byte pair encoding). Theo phương pháp mới này, hầu hết các từ đều có thể biểu diễn bởi subword và chúng ta sẽ hạn chế được một số lượng đáng kể các token. `<unk>` đại diện cho từ chưa từng xuất hiện trước đó. Rất nhanh chóng, Phương pháp mới đã được áp dụng ở hầu hết các phương pháp NLP hiện đại từ các lớp model BERT cho tới các biến thể của nó như OpenAI GPT, RoBERTa, DistilBERT, XLMNet. Kết quả áp dụng tokenize theo phương pháp mới đã cải thiện được độ chính xác trên nhiều tác vụ dịch máy, phân loại văn bản, dự báo câu tiếp theo, hỏi đáp, dự báo mối quan hệ văn bản.

Thuật toán BPE:

BPE (Byte Pair Encoding) là một kỹ thuật nén từ cơ bản giúp chúng ta index được toàn bộ các từ kể cả trường hợp từ mở (không xuất hiện trong từ điển) nhờ mã hóa các từ bằng chuỗi các từ phụ (subwords). Nguyên lý hoạt động của BPE dựa trên phân tích trực quan rằng hầu hết các từ đều có thể phân tích thành các thành phần con.

Chẳng hạn như từ: `low`, `lower`, `lowest` đều là hợp thành bởi `low` và những đuôi phụ `er`, `est`. Những đuôi này rất thường xuyên xuất hiện ở các từ. Như vậy khi biểu diễn từ `lower` chúng ta có thể mã hóa chúng thành hai thành phần từ phụ (subwords) tách biệt là `low` và `er`. Theo cách biểu diễn này sẽ không phát sinh thêm một index mới cho từ `lower` và đồng thời tìm được mối liên hệ giữa `lower`, `lowest` và `low` nhờ có chung thành phần từ phụ là `low`.

Phương pháp BPE sẽ thống kê tần suất xuất hiện của các từ phụ cùng nhau và tìm cách gộp chúng lại nếu tần suất xuất hiện của chúng là lớn nhất. Cứ tiếp tục quá trình gộp từ phụ cho tới khi không tồn tại các subword để gộp nữa, ta sẽ thu được tập subwords cho toàn bộ bộ văn bản mà mọi từ đều có thể biểu diễn được thông qua subwords.

Code của thuật toán BPE đã được tác giả chia sẻ tại subword-nmt.

Qúa trình này gồm các bước như sau:

- Bước 1: Khởi tạo từ điển (vocabulary).
- Bước 2: Biểu diễn mỗi từ trong bộ văn bản bằng kết hợp của các ký tự với token `<\w>` ở cuối cùng đánh dấu kết thúc một từ (lý do thêm token sẽ được giải thích bên dưới).
- Bước 3: Thống kê tần suất xuất hiện theo cặp của toàn bộ token trong từ điển.
- Bước 4: Gộp các cặp có tần suất xuất hiện lớn nhất để tạo thành một n-gram theo level character mới cho từ điển.
- Bước 5: Lặp lại bước 3 và bước 4 cho tới khi số bước triển khai merge đạt đỉnh hoặc kích thước kỳ vọng của từ điển đạt được.

Bạn sẽ dễ hình dung hơn qua ví dụ bên dưới:

Giả sử từ điển của chúng ta gồm các từ với tần suất như sau: `vocab = {'l o w </w>': 5, 'l o w e r </w>': 2, 'n e w e s t </w>': 6, 'w i d e s t </w>': 3}`.

Coi mỗi ký tự là một token. Khi đó thống kê tần suất xuất hiện của các cặp ký tự như sau:

```
import collections
```

https://phamdinhkhanh.github.io/2020/06/04/PhoBERT_Fairseq.html

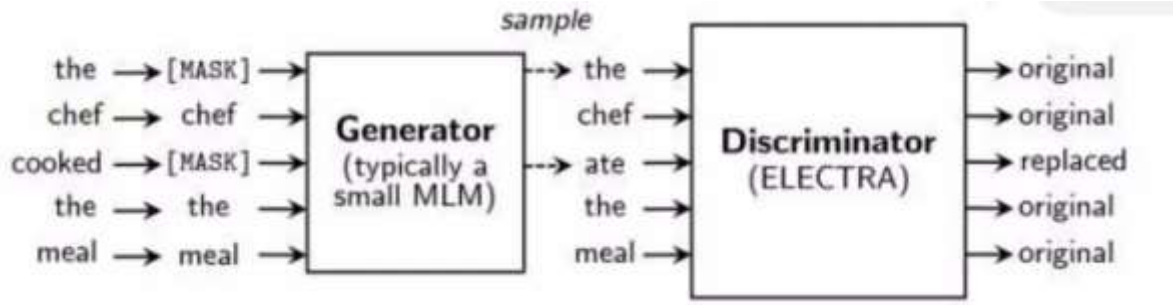
DEBERTA-V3

- Dựa theo **ELECTRA-style training** và thay thế phương pháp pre-training **MLM** trong DeBerta thành **Replace Token Detection (RTD)**.
- Gradient-disentangled embedding sharing **(GDES)**

DEBERTA-V3

ELECTRA-style training

- Không như Bert chỉ sử dụng 1 khối transformer encoder cho quá trình pre-training MLM, phương pháp ELECTRA sử dụng 2 khối như hình

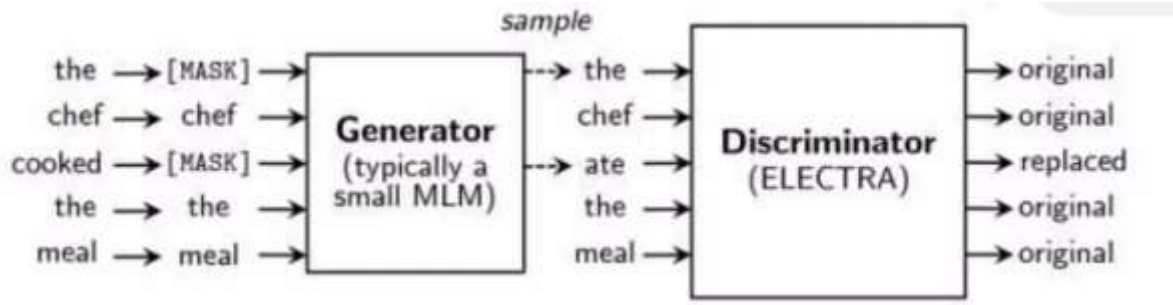


- Khối generator dùng cho MLM
- Khối discriminator nhận input từ khối generator và phân loại xem những token đó là token gốc hoặc token đã được thay thế.

DEBERTA-V3

ELECTRA-style training for Deberta-v3

- Trong mô hình deberta-v3, generator có độ sâu bằng 1 nửa so với discriminator

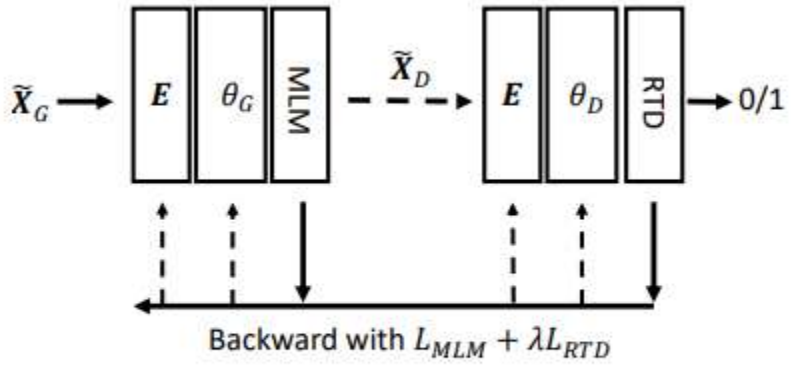


DEBERTA-V3

Gradient-disentangled embedding sharing (GDES)

TOKEN EMBEDDING SHARING IN ELECTRA

- Embedding Sharing **(ES)** cho phép **generator cung cấp inputs chứa nhiều thông tin** cho **discriminator** và **giảm số lượng tham số** cần học.
- Tuy nhiên pre-train như vậy sẽ gây ra vấn đề **multitask** và làm **chậm** quá trình **training** do trọng số bị chia sẻ với nhau gây ảnh hưởng đến quá trình học.
- Ngoài ra **MLM** và **RTD** có ảnh hưởng đối lập lên token embeddings. **MLM** làm cho các token embedding có ngữ nghĩa tương tự có khoảng cách gần nhau hơn. **RTD** thì cố gắng tách chúng ra để cho việc phân loại dễ hơn.



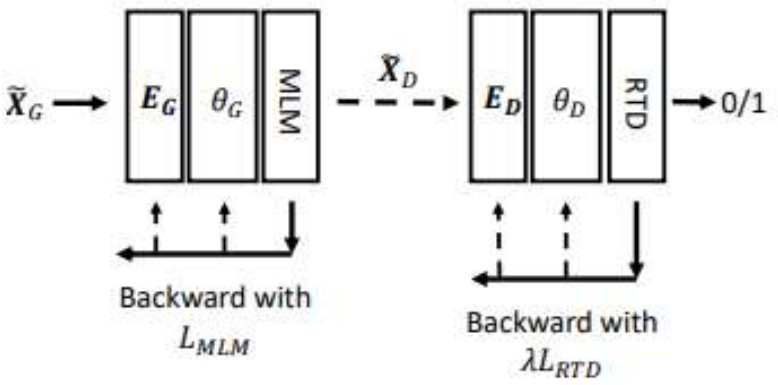
(a)

DEBERTA-V3

Gradient-disentangled embedding sharing (GDES)

TOKEN **No Embedding Sharing** IN ELECTRA

- No Embedding Sharing **(NES)** update para của MLM và RTD luân phiên nhau ở mỗi training step.
- Kết quả so sánh giữa **NES** và **ES:**
  + NES hội tụ nhanh hơn -> tránh được conflict giữa 2 task MLM và RTD làm chậm quá trình training
  + NES tạo ra được 2 model embedding riêng biệt -> MLM tạo ra được embedding mang thông tin chặt chẽ hơn RTD
- Tuy nhiên, kết quả khi sử dụng model này cho các task cụ thể không tăng đáng kể.



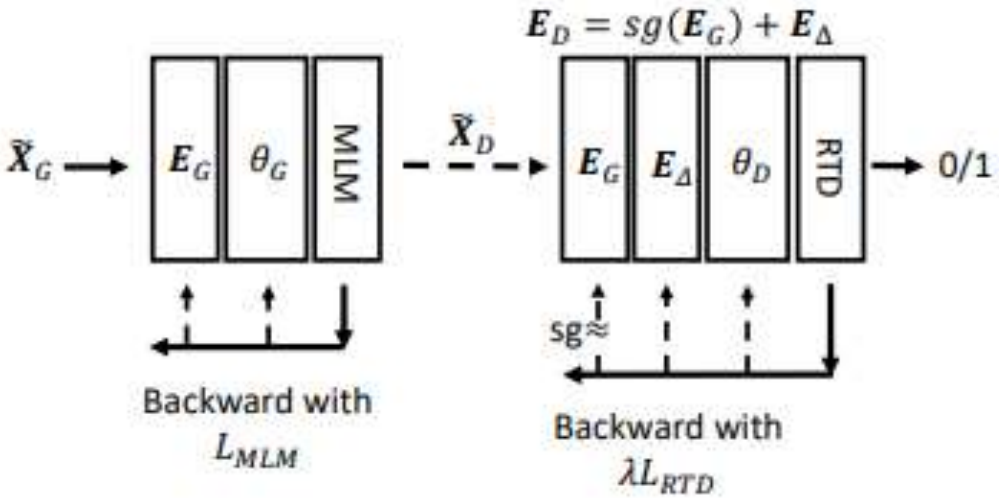
(b)

=> Điều này ủng hộ 1 giả thiết ES làm cho discriminator có thể lấy được thông tin hữu ích từ embeddings của generator.

DEBERTA-V3

Gradient-disentangled embedding sharing (GDES)

- GDES chia sẻ token embeddings như ES giúp mô hình học được nhiều thông tin được encode trong embeddings.
- Tuy nhiên GDES không sử dụng RTD loss để cập nhật trọng số cho generator.



Đánh giá

Nếu xét các class với label 0 (người viết) là positive thì ta có kết quả sau:
- Accuracy: 0.9598
- Precision: 0.9426
- Recall: 0.9947 => mô hình rất nhạy trong việc dự đoán các đoạn văn bản do người viết
- F1 Score: 0.9679

Nếu xét các class với label 1 (GPT) là positive thì ta có kết quả sau:
- Accuracy: 0.9598
- Precision: 0.9910 => mô hình đảm bảo được các đoạn văn bản mà mô hình cho rằng do GPT viết thực sự do chính GPT viết.
- Recall: 0.9052
- F1 Score: 0.9461

(1) https://www.geeksforgeeks.org/explanation-of-bert-model-nlp/
(2) Sh-tsang.medium.com
(3) DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing | PPT (slideshare.net)