**Báo cáo tiến trình học tập tuần 7**

1. **Byte Pair Encoding Tokenization (BPE algorithm)**

* Giả sử trong tập corpus có 5 từ sau : “hug”, “pug”, “pun”, “bun”, “hugs”’.
* Từ đó, ta có tập từ vựng cơ sở sẽ là [“b”, “g”, “h”, “n”, “p”, “s”, “u”].
* Thêm các token mới cho đến khi đạt được kích thước từ vựng mong muốn bằng cách hợp nhất (merge).
* Giả sử các từ trong tập corpus ban đầu có tần suất như sau:

("hug", 10), ("pug", 5), ("pun", 12), ("bun", 4), ("hugs", 5).

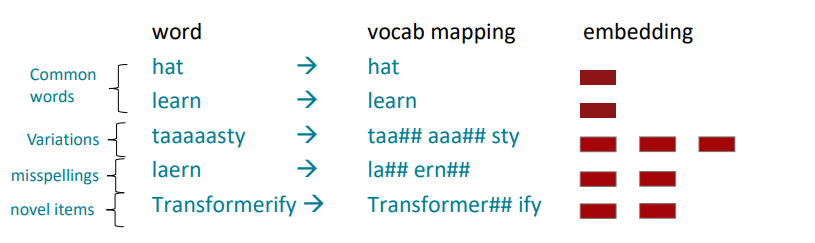
* Thuật toán chia các từ thành các kí tự:

("h" "u" "g", 10), ("p" "u" "g", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "u" "g" "s", 5).

* Cụm “ug” xuất hiện nhiều nhất với 20 lần sẽ được thêm vào từ vựng, và cặp từ vựng cũng sẽ được hợp nhất trong tất các các từ của corpus:



* Cứ tiếp tục như vậy đến số lượng từ vựng mong muốn.

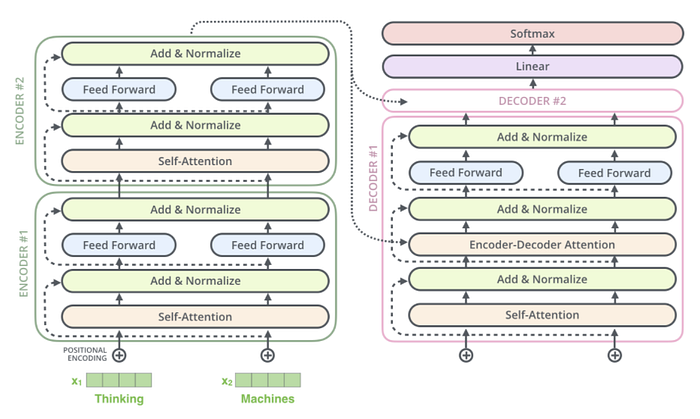


* Các kiến trúc như Transformer sẽ học các cụm ( như 3 vector embedding của taaaaasty) đứng cạnh nhau thường mang nghĩa giống “tasty”.

1. **Cấu trúc T5**
2. **Giới thiệu về T5**

* Thay vì thiết kế mô hình riêng cho mỗi tác vụ NLP (dịch, tóm tắt, phân loại …), T5 chuyển tất cả thành một bài toán duy nhất: **Text 🡪 Text.**
* Chỉ cần một mô hình cho nhiều tác vụ.

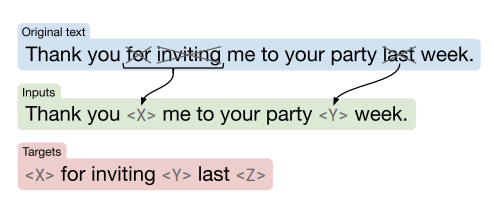
1. **Kiến trúc tổng quát**



* Kiến trúc T5 dựa trên Transformer encoder-decoder, khác với BERT (chỉ có encoder) và GPT (chỉ decoder), T5 dùng cả hai.

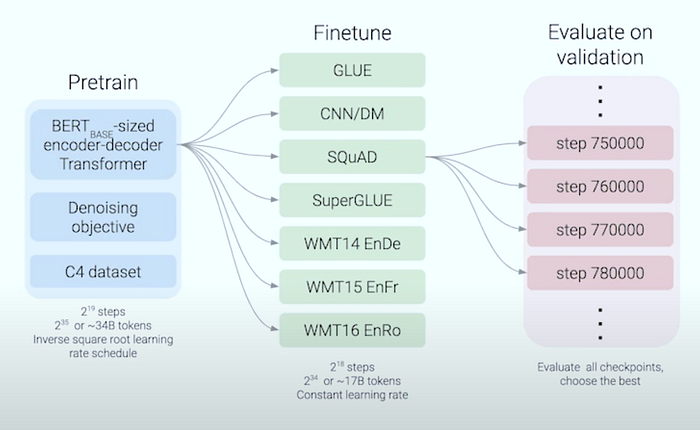
1. **Pretrained dataset**

* Encoder học representation mạnh, còn decoder học sinh chuỗi như LM, nhận thấy cách tốt nhất không phải LM thuần mà là **span corruption:**
  + Xóa bỏ (mask) một đoạn liên tiếp (span) trong câu.
  + Thay thế đoạn đó bằng một token đặc biệt như <x>.
  + Mục tiêu của decoder: sinh lại chính xác đoạn bị che đi.



* Ưu điểm: thay vì che lẻ tẻ từng từ (như BERT), T5 học cách sinh **cụm từ** (span) mạch lạc hơn.
* T5 huấn luyện trên tập C4 (Colossal Clean Crawled Corpus), được xây dựng từ Common Crawl.
* Kích thước: ~750GB text sạch (đã loại bỏ HTML, quảng cáo, duplicate…).
* Đây là một trong những tập dữ liệu khổng lồ nhất thời đó.

1. **Workflow**



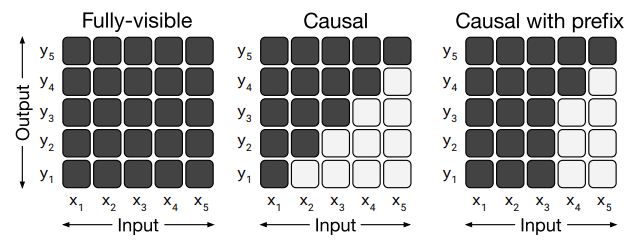
* **Pretrain** trên tập dữ liệu lớn.
* **Finetune** cho tưng task cụ thể (QA, summarization, translation, …), ban đầu thì người ta dùng promt và fine-tune toàn bộ tham số, rồi sao lưu tham số cho mỗi task.
* **Evaluate :**
  + Trong quá trình fine-tuning, mô hình được lưu checkpoint nhiều lần (vd: step 750k, 760k, 770k, 780k…).
  + **Đánh giá tất cả checkpoints trên tập validation** → chọn mô hình tốt nhất, không phải cứ checkpoint cuối cùng là tối ưu.

1. **So sánh**

* Trong quá trình pretraining, có hai yếu tố quyết định chất lượng mô hình:
  + Kiến trúc (cách tổ chức attention, encoder/decoder)
  + Mục tiêu huấn luyện (objective – mô hình được yêu cầu dự đoán gì).

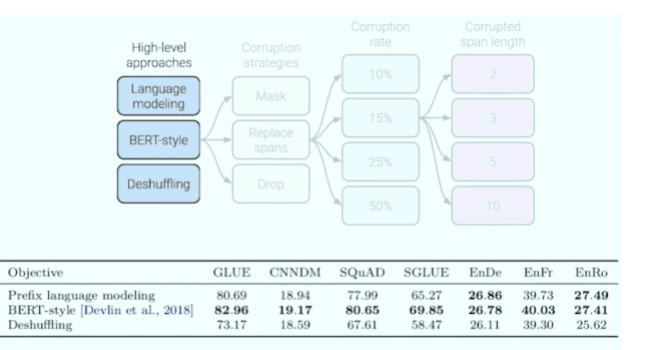
1. **Kiến trúc**

* Có 3 hướng chính:
  + **Fully-visible mask (BERT):** input có thể nhìn toàn bộ, nhưng không sinh chuỗi output.
  + **Causal mask (GPT):** sinh chuỗi autoregressive, nhưng không tận dụng được ngữ cảnh hai chiều.
  + **Prefix LM (Encoder–Decoder, T5):** encoder nhìn toàn bộ input (bidirectional), decoder sinh output theo causal mask.

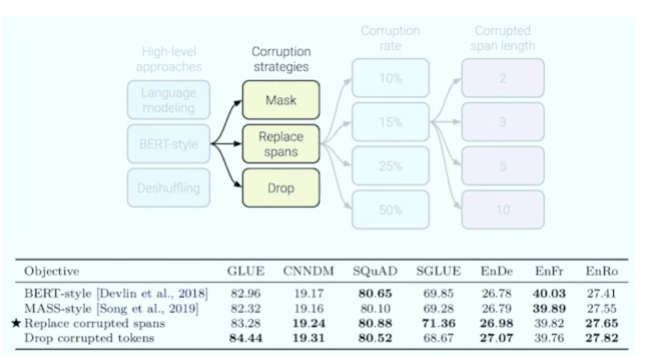


1. **Mục tiêu huấn luyện**

* Có 3 cách tiếp cận:
  + **Language Modeling (LM):** dự đoán từ tiếp theo dựa trên ngữ cảnh trước đó.
  + **BERT-style:** che (mask) một số token hoặc cụm từ trong câu rồi dự đoán lại.
  + **Deshuffling:** xáo trộn trật tự các từ và yêu cầu mô hình khôi phục văn bản gốc.

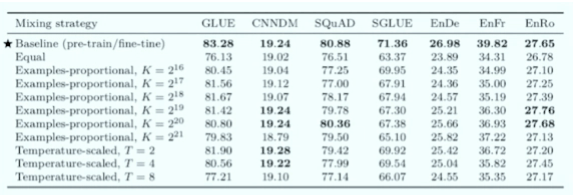
****

* Có 3 lựa chọn để corruption là:
  + **Mask:** Che từng từ riêng lẻ
  + **Replace spans:** Thay thế cả một đoạn liên tiếp của input bàng một sentinel token đặc biệt.
  + **Drop:** bỏ luôn một phần, mô hình phải tự suy ra phần bị mất.

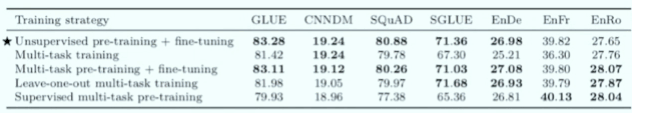
****

1. **Multi-task**

* Áp dụng các cách khác nhau để trộn dữ liệu:
  + **Equal →** mỗi task được lấy số lượng mẫu bằng nhau, bất kể dataset to nhỏ.
  + **Examples-proportional →** lấy mẫu tỉ lệ thuận với kích thước dataset.
  + **Temperature-scaled sampling →** điều chỉnh tỉ lệ bằng nhiệt độ (T).
    - Ý tưởng: thay vì lấy đúng tỉ lệ, ta nâng lũy thừa theo công thức:

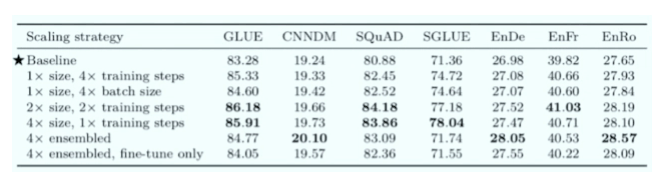


* Các chiến lược training:
  + **Multi-task training:** sau khi mix dữ liệu, huấn luyện nhiều task cùng lúc.
  + **Multi-task pretraning + fine-tuning:** đầu tiên pretrain trên hỗn hợp nhiều task, sau đó fine-tune riêng cho từng task.
  + **Leave-one-out multi-task training:** Pretrain trên hỗn hợp các task không chứa task mục tiêu (ví dụ pretrain trên A và C, không có B), sau đó fine-tune trên task bị loại ra.
  + **Supervised multi-task pre-training:** Chỉ pretrain trên các dataset supervised (không dùng dữ liệu unsupervised khổng lồ), rồi fine-tune.



1. **Scaling**

* Khi model to hơn và train lâu hơn thì kết quả tăng đáng kể.

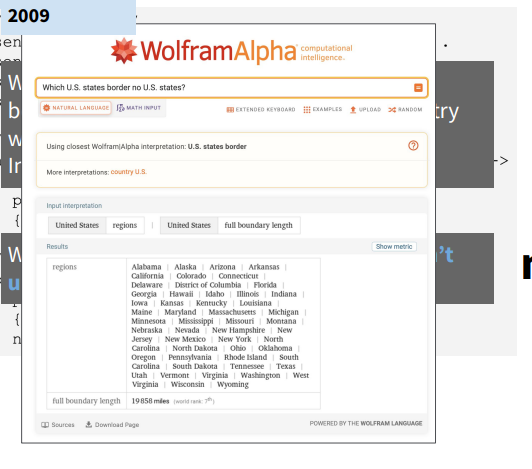
****

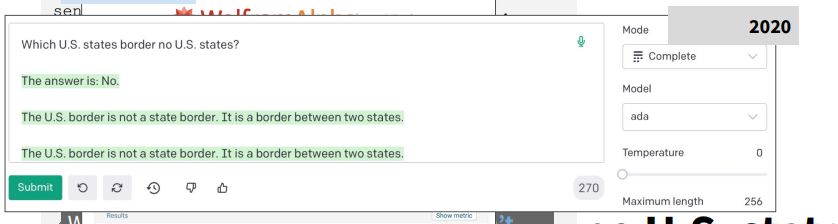
1. **Giới thiệu và sự phát triển của Natural Language Understanding (NLU)**

* **Natural Language Understanding (NLU)** là một nhánh quan trọng của **Xử lý ngôn ngữ tự nhiên (NLP)** tập trung vào việc giúp máy tính **hiểu được ý nghĩa, ngữ cảnh và ý định** ẩn sau ngôn ngữ của con người.
* Sự phát triển của các mô hình AI từ 1980 đến nay cho thấy sự tiến bộ vượt bậc trong khả năng hiểu và xử lý:





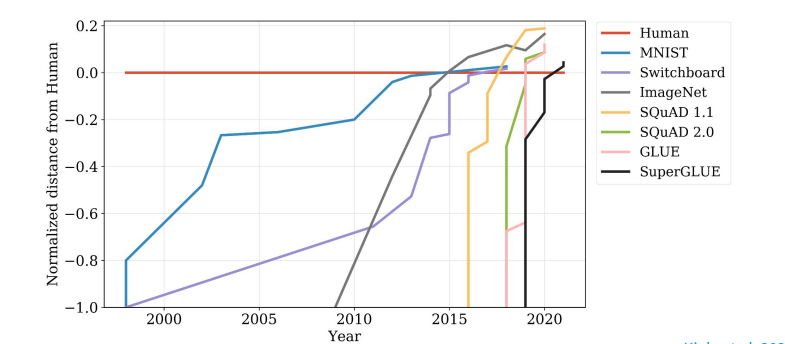




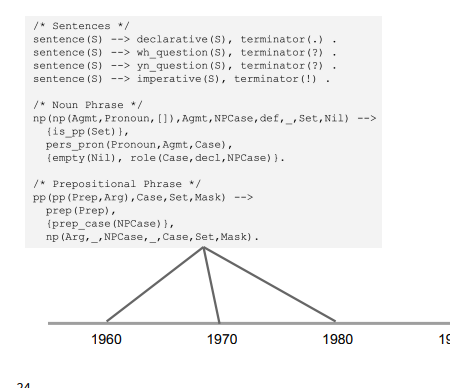


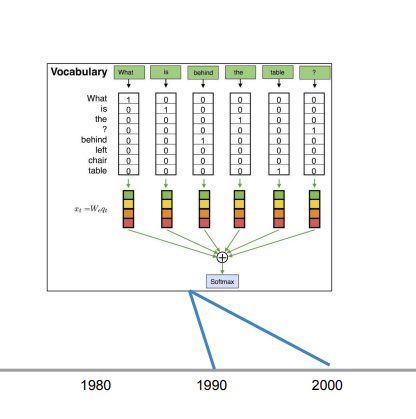


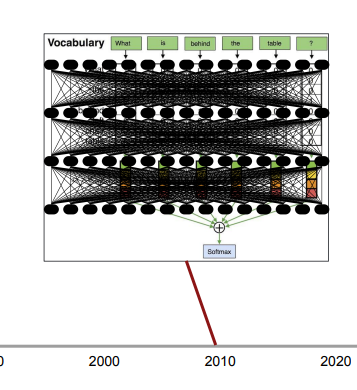
* Các model ngày càng vượt benchmark quá nhanh:

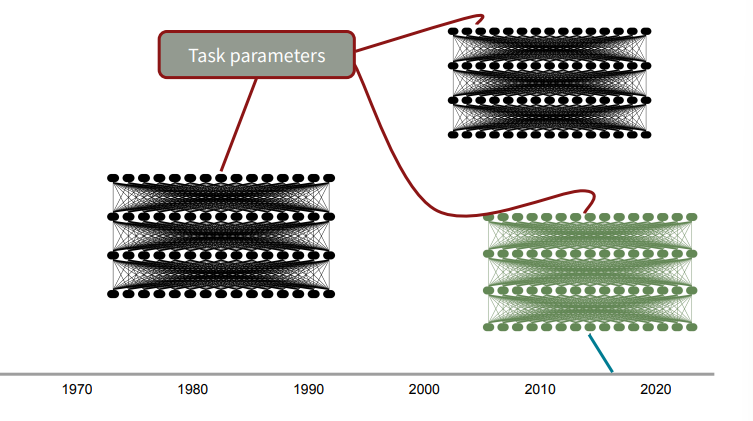


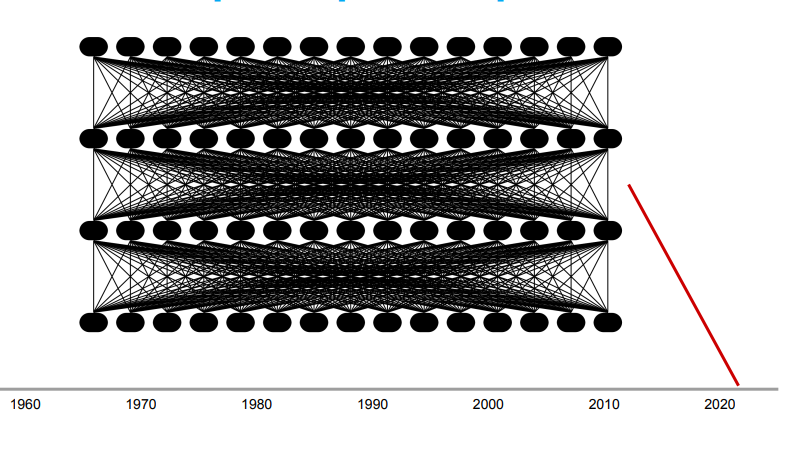
* Các model AI qua các thời kì:



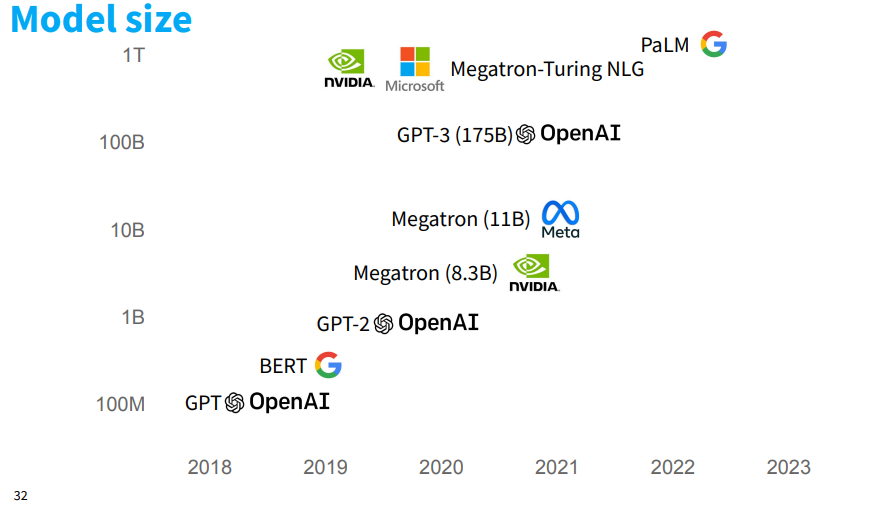






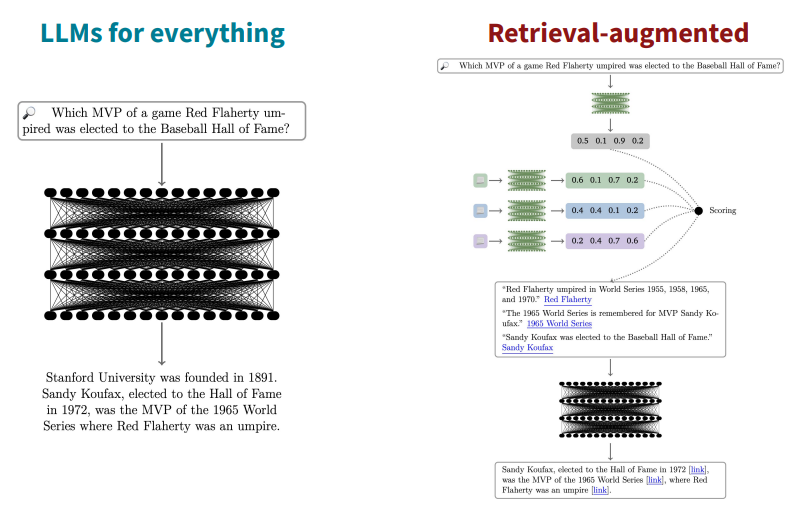


* Sự phát triển của các model:

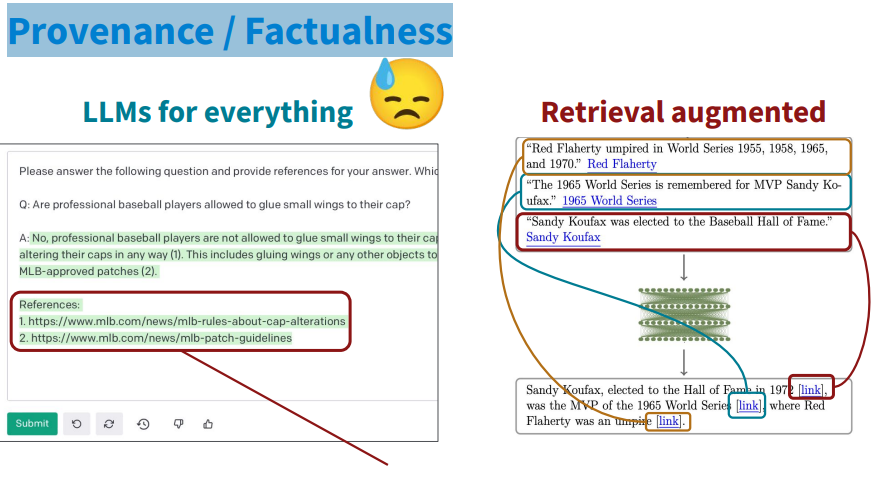


* Các bài báo liên quan đến biểu diễn dưới dạng vector lần lượt xuất hiện, mở đầu là Word2Vec, Glove, sau đó là ELMo (Embeddings from Language Models, bước đột phá lớn), sau đó xuất hiện BERT và GPT.
* ChatGPT học từ phản hồi của con người bằng cách:
  + **Bước 1:** Cho mô hình biết cách trả lời cơ bản bằng supervised learning.
  + **Bước 2:** Xây dựng thước đo “câu trả lời tốt” nhờ con người.
  + **Bước 3:** Dùng RL để tinh chỉnh mô hình sao cho tối đa hóa sự “hài lòng” của con người.

1. **Retrieval-augmented in-context learning**

****

* **Độ hiệu quả: Retrieval-augmented + smaller LMs** hiệu quả, tiết kiệm, dễ triển khai, mà vẫn giải quyết được bài toán nhờ tận dụng tìm kiếm ngoài.
* **Updateability: Retrieval-augmented** được cập nhật các thông tin mới.
* **Nguồn thông tin:**



* **An toàn và bảo mật:**
  + LLMs ghi nhơcs nguyên văn dữ liệu huấn luyện, bao gồm các dữ liệu nhạy cảm. LLMs khó kiểm soát quyền truy cập, nếu mở quyền cho một người, họ có thể truy cập toàn bộ năng lực của model.
  + Vì kiến thức chính được lấy từ **retrieval corpus (database, tài liệu, search index)** thay vì nằm sẵn trong weights của LLM → ta có thể áp dụng các **Access restriction** (quyền hạn truy cập) **ngay trên tài liệu**.
* Điểm yếu chết người của finetune là: khi mình finetune thì không đảm bảo model sẽ quên kiến thức cũ, và chi phí để finetune là không hề nhỏ.
* rag có một vấn đề là promt đầu vào của nó rất dài, mà self-attention tăng theo số mũ, làm cho chi phí tính toán cực kì lớn. Với việc finetune lại thì chi phí tính toán để quản lí data để truy vấn thì phần quản lí data có chi phí tính toán lớn hơn rất nhiều