TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TRẦN VĂN THẢO - 52000805**

**NGUYỄN VÕ CÔNG HUY - 52000765**

**XÂY DỰNG HỆ THỐNG TƯ VẤN TUYỂN SINH CHO SINH VIÊN**

**DỰ ÁN CÔNG NGHỆ THÔNG TIN**

**TÍNH TOÁN THÔNG MINH**

**VÀ HỆ THỐNG THÔNG TIN**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TRẦN VĂN THẢO - 52000805**

**NGUYỄN VÕ CÔNG HUY - 52000765**

**XÂY DỰNG HỆ THỐNG TƯ VẤN TUYỂN SINH CHO SINH VIÊN**

**DỰ ÁN CÔNG NGHỆ THÔNG TIN**

**TÍNH TOÁN THÔNG MINH**

**VÀ HỆ THỐNG THÔNG TIN**

Người hướng dẫn

**TS. TRẦN LƯƠNG QUỐC ĐẠI**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

**LỜI CẢM ƠN**

Lời nói đầu tiên, chúng em xin gửi lời cảm ơn chân thành nhất, đặc biệt nhất đến ***TS Trần Lương Quốc Đại*** đã hỗ trợ và truyền đạt những kiến thức quý báo cũng như đã tạo mọi điều kiện thuận lợi nhất để chúng em được tiếp cận và hoàn thành môn Dự án công nghệ thông tin.

Với thời gian và trình độ còn hạn chế, bài báo cáo không thể tránh khỏi những thiếu sót. Kính mong thầy sẽ chỉ bảo cũng như là đóng góp ý kiến để bài báo cáo của chúng em ngày càng dần hoàn thiện hơn. Đó sẽ là hành trang quý giá để chúng em có thể hoàn thiện về kĩ năng và kiến thức liên quan đến những vấn đề sau này.

Lời cuối cùng, chúng em xin kính chúc thầy thật nhiều sức khỏe, thành công, hạnh phúc và luôn giữ mãi sự nhiệt huyết để có thể giúp thêm thật nhiều thế hệ sinh viên Trường Đại Học Tôn Đức Thắng có nhiều sự tự tin và vững kiến thức với môn Dự án công nghệ thông tin. Ngọn lửa của môn học sẽ ngày càng được lan tỏa rộng rãi nhiều hơn!

*TP. Hồ Chí Minh, ngày 03 tháng 03 năm 2024*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

*N.V.C Huy*

*Nguyễn Võ Công Huy*

*T.V Thảo*

*Trần Văn Thảo*

**CÔNG TRÌNH ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi và được sự hướng dẫn khoa học của TS. Trần Lương Quốc Đại. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong Dự án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung Dự án của mình**. Trường Đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 03 tháng 03 năm 2024*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

*N.V.C Huy*

*Nguyễn Võ Công Huy*

*T.V Thảo*

*Trần Văn Thảo*

**XÂY DỰNG HỆ THỐNG TƯ VẤN TUYỂN SINH CHO SINH VIÊN**

**TÓM TẮT**

Đề tài "Xây dựng hệ thống Tư vấn Tuyển sinh cho Sinh viên" nhấn mạnh việc tận dụng trí tuệ nhân tạo để cung cấp thông tin chi tiết và hữu ích về quy trình tuyển sinh. Sử dụng các mô hình như BiLSTM, PhoBERT, GPT-2 và MRCQuestionAnswering , đề tài hướng tới việc cải thiện khả năng trả lời câu hỏi và đưa ra thông tin đa dạng. Hệ thống web được xây dựng tích hợp giao diện thân thiện để tương tác với người dùng, cung cấp thông tin về tra cứu tuyển sinh và hỗ trợ đưa ra quyết định thông tin và chi tiết cho sinh viên. Đánh giá hiệu suất của mỗi mô hình và đề xuất cải tiến cho tương lai là các phần quan trọng trong quá trình nghiên cứu và phát triển của đề tài này.

**STUDENT ADMISSION ADVISORY SYSTEM**

**ABSTRACT**

The research topic "Student Admission Advisory System" emphasizes leveraging artificial intelligence to provide detailed and useful information about the admissions process. Utilizing models such as BiLSTM, PhoBERT, GPT-2, and MRCQuestionAnswering , the project aims to enhance the ability to answer questions and provide diverse information. The web system is constructed with a user-friendly interface for interacting with users, offering information on admissions queries, and assisting in decision-making for students. Evaluating the performance of each model and proposing improvements for the future are crucial components of the research and development process for this project.

**MỤC LỤC**

[DANH MỤC HÌNH VẼ vii](#_Toc161244936)

[DANH MỤC BẢNG BIỂU viii](#_Toc161244937)

[DANH MỤC CÁC CHỮ VIẾT TẮT ix](#_Toc161244938)

[CHƯƠNG 1. MỞ ĐẦU VÀ TỔNG QUAN ĐỀ TÀI 1](#_Toc161244939)

[1.1 Lý do chọn đề tài 1](#_Toc161244940)

[1.2 Mục tiêu thực hiện đề tài 1](#_Toc161244941)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 2](#_Toc161244942)

[2.1 Hệ thống hỏi đáp mở 2](#_Toc161244943)

[2.1.1 Hệ thống hỏi đáp mở là gì? 2](#_Toc161244944)

[2.1.2 Các loại hệ thống hỏi đáp mở trong ODQA 2](#_Toc161244945)

[2.1.3 Các mô hình hỏi đáp mở 3](#_Toc161244946)

[2.1.4 Lý thuyết các mô hình được sử dụng trong đề tài 8](#_Toc161244947)

[2.2 Module Website 29](#_Toc161244948)

[2.2.1 MongoDB và Mongoose 29](#_Toc161244949)

[2.2.2 ExpressJS 30](#_Toc161244950)

[2.2.3 ReactJS 31](#_Toc161244951)

[2.2.4 NodeJS 32](#_Toc161244952)

[CHƯƠNG 3. MÔ HÌNH ĐỀ XUẤT 34](#_Toc161244953)

[3.1 Các mô hình kết hợp 34](#_Toc161244954)

[3.1.1 BM-25 - BiLSTM 34](#_Toc161244955)

[3.1.2 BM-25 - MRCQuestionAnswering 34](#_Toc161244956)

[3.1.3 BM-25 - PhoBERT 34](#_Toc161244957)

[3.2 Mô hình không kết hợp 34](#_Toc161244958)

[3.2.1 GPT-2 34](#_Toc161244959)

[CHƯƠNG 4. THỰC NGHIỆM 36](#_Toc161244960)

[4.1 Dữ liệu thực nghiệm 36](#_Toc161244961)

[4.2 Cài đặt thực nghiệm 36](#_Toc161244962)

[4.2.1 Chuẩn bị Môi trường và cài đặt thư viện 36](#_Toc161244963)

[4.2.2 Xây dựng mô hình 36](#_Toc161244964)

[4.3 Kết quả thực nghiệm 41](#_Toc161244965)

[4.3.1 Kết quả so sánh điểm ROUGE 41](#_Toc161244966)

[4.3.2 Kết quả so sánh điểm BLEU 41](#_Toc161244967)

[CHƯƠNG 5. KẾT QUẢ ĐẠT ĐƯỢC 42](#_Toc161244968)

[5.1 EndUser website module 42](#_Toc161244969)

[5.1.1 Màn hình trang chủ 42](#_Toc161244970)

[CHƯƠNG 6. KẾT LUẬN 44](#_Toc161244971)

[6.1 Kết luận 44](#_Toc161244972)

[6.2 Hướng phát triển 44](#_Toc161244973)

[TÀI LIỆU THAM KHẢO 45](#_Toc161244974)

# DANH MỤC HÌNH VẼ

# DANH MỤC BẢNG BIỂU

# DANH MỤC CÁC CHỮ VIẾT TẮT

|  |  |
| --- | --- |
| ODQA | Open Domain Question Answering |
| IR | Information Retrieval |
| TF-IDF | Term Frequency-Inverse Document Frequency |
| BM-25 | Best Match 25 |
| RNN | Recurrent Neural Network |
| LSTM | Long Short-term Memory |
| BiLSTM | Bidirectional Long Short-term Memory |
| BERT | Bidirectional Encoder Representations from Transformers |
| BPE | Byte Pair Encoding |
| LLMs | Large Language Models |
| MLM | Masked Language Model |
| NLP | Natural Language Processing |
| NSP | Next Sentence Prediction |
| GPT | Generative Pre-trained Transformers |
| MRC | Machine Reading Comprehension |
| FAIR | Facebook AI Research |

# MỞ ĐẦU VÀ TỔNG QUAN ĐỀ TÀI

## Lý do chọn đề tài

Một trong những lý do quan trọng để chọn đề tài này là đáp ứng nhu cầu ngày càng tăng về thông tin trong quá trình tuyển sinh. Hệ thống tuyển sinh truyền thống thường gặp khó khăn trong việc cung cấp thông tin nhanh chóng và chính xác đối với các học sinh đang tìm hiểu về quy trình tuyển sinh. Sự chậm trễ trong phản hồi có thể tạo ra rủi ro về việc học sinh không nhận được thông tin đầy đủ và kịp thời, ảnh hưởng đến quyết định của họ trong việc chọn lựa trường học.

Hệ thống Q&A có khả năng ngay lập tức cung cấp thông tin chi tiết và rõ ràng, giúp học sinh giải đáp mọi thắc mắc một cách hiệu quả. Bằng cách này, đề tài không chỉ giải quyết vấn đề bất cập trong quá trình tuyển sinh mà còn đáp ứng nhu cầu ngày càng cao của học sinh đối với thông tin nhanh chóng và dễ tiếp cận.

## Mục tiêu thực hiện đề tài

Mục tiêu chính của nghiên cứu là phát triển một hệ thống Q&A trong quá trình tuyển sinh, nhằm giảm thời gian phản hồi, đa dạng hóa khả năng trả lời, tự động cập nhật thông tin, và thực nghiệm ứng dụng trong môi trường thực tế. Mục tiêu này nhằm tối ưu hóa trải nghiệm thông tin cho học sinh, giảm công sức của tổ chức tuyển sinh, và tạo cơ sở cho phát triển tương lai trong ứng dụng trí tuệ nhân tạo trong giáo dục.

# CƠ SỞ LÝ THUYẾT

## Hệ thống hỏi đáp mở

### Hệ thống hỏi đáp mở là gì?

Hệ thống hỏi đáp mở (Open Domain Question Answering) là một loại nhiệm vụ ngôn ngữ tự nhiên, yêu cầu mô hình tạo ra câu trả lời cho các câu hỏi đối với dữ liệu thực tế bằng ngôn ngữ tự nhiên. Câu trả lời đúng là khách quan nên việc đánh giá hiệu quả của mô hình rất đơn giản.

Ví dụ:

Câu hỏi: Khi nào bắt đầu thời gian đăng ký xét tuyển đại học năm nay?

Câu trả lời: Thời gian đăng ký xét tuyển đại học năm nay bắt đầu từ ngày 15 tháng 6 đến hết ngày 30 tháng 6.

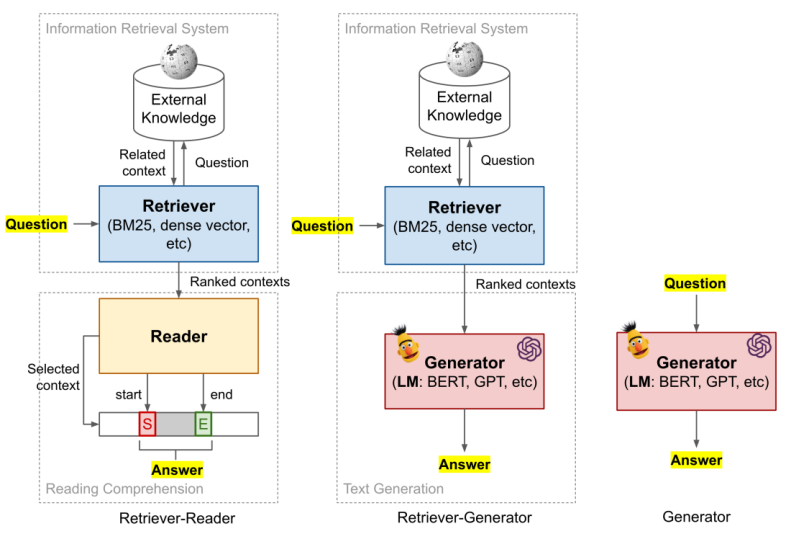
Phần "open-domain" đề cập đến sự thiếu hụt của ngữ cảnh liên quan đối với bất kỳ câu hỏi thực tế nào được đặt ra một cách tùy ý. Trong ví dụ trên, mô hình chỉ nhận câu hỏi làm đầu vào mà không có bất kỳ thông tin nào về "tại sao quyết định đăng ký xét tuyển lại có thời gian như vậy" được cung cấp. Trong trường hợp cả câu hỏi và ngữ cảnh đều được cung cấp, nhiệm vụ này được biết đến là Đọc hiểu (Reading Comprehension - RC).

Một mô hình ODQA có thể hoạt động có hoặc không có quyền truy cập vào nguồn tri thức bên ngoài (ví dụ: trang web của trường đại học), và điều này có thể được gọi là open-book hoặc closed-book.

### Các loại hệ thống hỏi đáp mở trong ODQA

Khi xem xét các loại hỏi đáp trong open-domain, thường được phân loại theo 3 loại có độ khó tăng dần:

* Mô hình có khả năng ghi nhớ đúng và trả lời với câu trả lời của một câu hỏi mà nó đã thấy trong quá trình huấn luyện.
* Mô hình có khả năng trả lời các câu hỏi mới trong quá trình kiểm thử và chọn câu trả lời từ tập các câu trả lời mà nó đã thấy trong quá trình huấn luyện.
* Mô hình có khả năng trả lời các câu hỏi mới mà có câu trả lời không xuất hiện trong tập dữ liệu huấn luyện.



Hình 2.1 Tổng quan về ba loại mô hình hỏi đáp mở

### Các mô hình hỏi đáp mở

#### Retriever – Reader

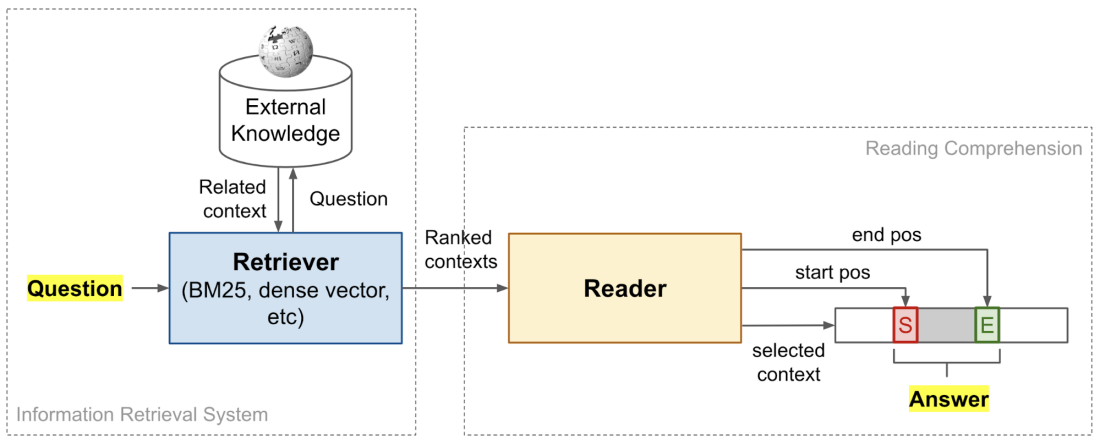
1. Tổng quan về quy trình thực hiện

Dựa trên một câu hỏi thông tin cụ thể, nếu một mô hình ngôn ngữ không có ngữ cảnh hoặc không đủ lớn để ghi nhớ ngữ cảnh có sẵn trong bộ dữ liệu huấn luyện, khả năng đoán đúng câu trả lời là khá thấp. Trong kỳ thi cho phép xem tài liệu, sinh viên được phép tham khảo các nguồn tài liệu bên ngoài như ghi chú và sách khi trả lời các câu hỏi kiểm tra. Tương tự, hệ thống Hỏi-Đáp mở (ODQA) có thể được kết hợp với một cơ sở kiến thức phong phú để xác định các tài liệu liên quan làm bằng chứng cho câu trả lời.

Chúng ta có thể phân tích quy trình này thành hai giai đoạn:

* Tìm kiếm ngữ cảnh liên quan trong một kho dữ liệu kiến thức bên ngoài (Retriever):
  + Ở giai đoạn này, hệ thống cần xác định các tài liệu, văn bản hoặc nguồn thông tin ngoại vi chứa thông tin liên quan đến câu hỏi.
  + Quá trình này có thể thực hiện thông qua một hệ thống hỏi đáp mở (ODQA) được kết hợp với một kho dữ liệu lớn chứa kiến thức đa dạng.
* Xử lý ngữ cảnh đã tìm được để trích xuất câu trả lời (Reader):
  + Sau khi có được ngữ cảnh liên quan, hệ thống cần phân tích và xử lý thông tin để trích xuất câu trả lời đáp ứng câu hỏi.
  + Các phương pháp xử lý ngôn ngữ tự nhiên và trích xuất thông tin có thể được áp dụng để hiểu và tìm ra câu trả lời trong ngữ cảnh tìm kiếm.

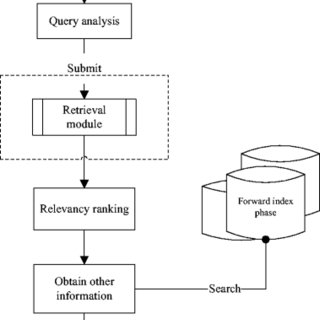
Để hiểu tổng quan về quy trình tìm kiếm và cho ra câu trả lời chúng ta có thể quan sát hình 2.2



Hình 2.2 Quy trình tìm kiếm câu trả lời cho câu hỏi của Retriever-Reader

1. Hệ thống thu thập thông tin (Retriever)

Hệ thống thu thập thông tin (Retriever) trong ngữ cảnh truy vấn thông tin (Information Retrieval) là một phần quan trọng của mô hình hệ thống truy vấn thông tin. Nhiệm vụ chính của Retriever là tìm kiếm và trả về các tài liệu phù hợp nhất với truy vấn từ người dùng.



Hình 2.3 Sơ đồ quy trình hệ thống thông tin hoạt động

Có hai phương pháp chính để triển khai hệ thống thu thập thông tin (retriever) trong hệ thống trả lời câu hỏi mở (ODQA):

* Hệ thống thu thập thông tin (IR) dựa trên các đặc trưng TF-IDF cổ điển không dựa trên học máy (Classic IR): Phương pháp này tập trung vào việc sử dụng hệ thống thu thập thông tin (IR) truyền thống, dựa trên đặc trưng TF-IDF để xác định sự tương đồng giữa câu hỏi và các văn bản nguồn. Các truy vấn được biểu diễn bằng các vector dựa trên tần suất xuất hiện của các từ khóa và các văn bản được sắp xếp theo độ tương đồng.
* Hệ thống thu thập thông tin (IR) dựa trên các vector dense của văn bản được tạo ra bởi các mạng nơ-ron (neural IR): Phương pháp này sử dụng các mô hình mạng nơ-ron để tạo ra các biểu diễn vector dense của văn bản, thường thông qua các mô hình như BERT, RoBERTa, hay các biến thể khác của Transformer. Các biểu diễn này mang lại khả năng hiểu ngữ nghĩa sâu rộng và tương đồng ngữ cảnh giữa câu hỏi và nguồn thông tin.

1. Quá trình đọc và trả lời (Reader)

Mô hình Reader là một phần quan trọng của hệ thống hỏi đáp mở, chịu trách nhiệm đọc và hiểu nội dung của văn bản để tìm kiếm thông tin cần thiết để trả lời các câu hỏi từ người dùng. Mục tiêu chính của mô hình là hiểu và rút trích thông tin ý nghĩa từ văn bản nguồn để tạo ra câu trả lời chính xác.

Kiến trúc chung của mô hình Reader gồm các phần sau:

* Lớp Embedding: Chuyển đổi từ ngôn ngữ tự nhiên thành các vector số thực để máy tính có thể hiểu được ý nghĩa của từng từ.
* Kiến Trúc Tự Chú Ý (Self-Attention): Các lớp tự chú ý giúp mô hình tập trung vào các phần quan trọng của văn bản, xử lý thông tin từ cấp từ đến cấp câu.
* Trích Xuất Thông Tin: Sau khi đọc và hiểu, mô hình trích xuất thông tin quan trọng để sử dụng trong quá trình tạo câu trả lời.

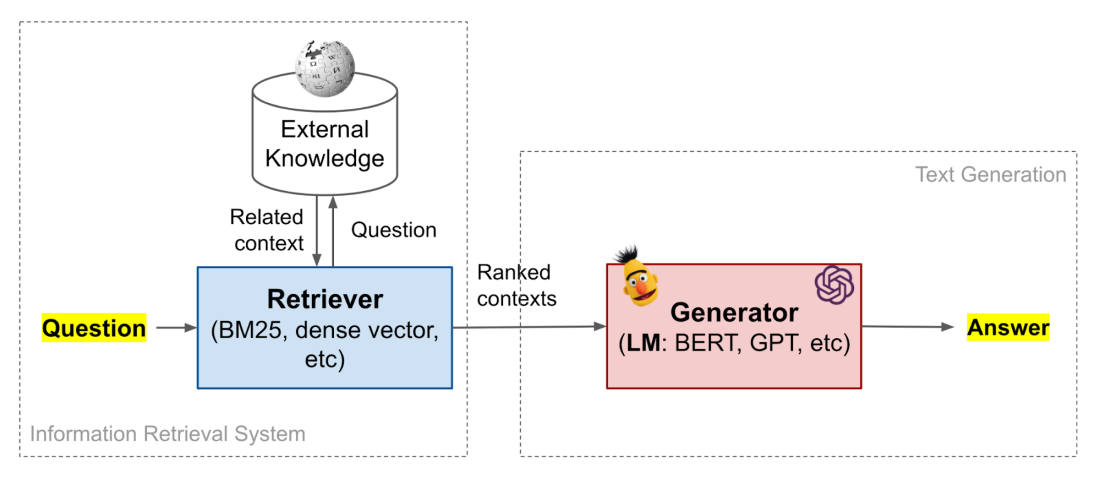
Các mô hình thường sử dụng trong vai trò Reader: Bidirectional LSTM, BERT,...

#### Retriever – Generative

1. Tổng quan về quy trình thực hiện

So với phương pháp thu thập thông tin và đọc hiểu (retriever-reader), phương pháp thu thập thông tin và tạo văn bản (retriever-generator) cũng gồm 2 giai đoạn, nhưng giai đoạn thứ hai được thiết kế để tạo ra văn bản tự do trực tiếp để trả lời câu hỏi thay vì trích xuất vị trí end/start trong một đoạn văn được thu thập. Một số nghiên cứu cũng gọi phương pháp này là "Generative Question Answering".

Để hiểu tổng quan về quy trình tìm kiếm và cho ra câu trả lời của retriever - generator chúng ta có thể quan sát hình 2.4



Hình 2.4 Quy trình tìm kiếm câu trả lời cho câu hỏi của Retriever-Generator

1. Quá trình tạo câu trả lời (Generator)

Không giống như mô hình Reader, mô hình Generator trong hệ thống hỏi đáp mở là một thành phần quan trọng giúp tạo ra câu trả lời chính xác và có ý nghĩa cho các câu hỏi mà hệ thống không có sẵn câu trả lời trong nguồn dữ liệu cố định.

Kiến trúc chung của mô hình Generator gồm các phần sau:

* Mạng Nơ-ron Tự Sinh (Seq2Seq): Sử dụng mô hình Seq2Seq, trong đó có một mạng nơ-ron Encoder chịu trách nhiệm biểu diễn câu hỏi đầu vào thành một vector ngữ cảnh. Mạng nơ-ron Decoder sẽ sinh ra câu trả lời dựa trên vector ngữ cảnh đã biểu diễn.
* Attention Mechanism: Các mô hình thường sử dụng cơ chế chú ý để tập trung vào các phần quan trọng của câu hỏi và nguồn dữ liệu khi tạo ra câu trả lời. Cơ chế này giúp mô hình tập trung vào các từ quan trọng và mối quan hệ trong ngữ cảnh.
* Transfer Learning và Pre-training: Sử dụng kỹ thuật pre-training để học các biểu diễn ngôn ngữ tự nhiên từ dữ liệu lớn trước khi fine-tuning cho nhiệm vụ ODQA cụ thể. Mô hình có thể sử dụng các kiến thức từ các nguồn lớn như Wikipedia để cải thiện khả năng sinh câu trả lời.

Các mô hình thường sử dụng trong vai trò Generator: T5, BERT,...

#### Generative

1. Tổng quan về quy trình thực hiện

Mô hình ngôn ngữ lớn (LLMs) trong ODQA, khi đối mặt với câu hỏi cụ thể, gặp khó khăn nếu không có đủ ngữ cảnh từ tập dữ liệu huấn luyện. Phương pháp sinh (generative approach) trong ODQA giúp mô hình không chỉ trích xuất thông tin mà còn sáng tạo ra câu trả lời tự nhiên và logic, tận dụng cơ sở kiến thức để làm bằng chứng cho câu trả lời.

1. Quá trình tạo câu trả lời (Generator)

Mô hình ngôn ngữ lớn (LLMs) sử dụng các phần trích xuất quan trọng và kiến thức đã nắm bắt từ ngữ cảnh câu hỏi để sinh ra câu trả lời. Quá trình này không chỉ là việc đơn giản là trích xuất thông tin, mà còn bao gồm các kỹ thuật sinh động để tạo ra câu trả lời tự nhiên và phù hợp.

Sử dụng các trích xuất quan trọng: Mô hình xác định và sử dụng các đoạn văn bản quan trọng (salient spans) từ nguồn dữ liệu để đảm bảo rằng câu trả lời chứa những thông tin quan trọng nhất.

Kỹ thuật ghép cắt và tái cấu trúc: Mô hình áp dụng các kỹ thuật ghép cắt thông tin và tái cấu trúc câu để tạo ra câu trả lời có logic và tự nhiên. Điều này bao gồm việc sắp xếp và kết hợp thông tin từ các đoạn văn bản khác nhau để tạo ra câu trả lời toàn diện.

Tích hợp các kiến thức Từ Pre-training: Mô hình sử dụng kiến thức đã học từ quá trình pre-training để cải thiện khả năng sinh câu trả lời. Điều này có thể bao gồm sử dụng thông tin từ nguồn dữ liệu lớn như Wikipedia, giúp đảm bảo câu trả lời phản ánh sự hiểu biết sâu sắc về ngôn ngữ tự nhiên.

Các mô hình thường sử dụng trong vai trò Generator: T5, GPT, BERT,...

### Lý thuyết các mô hình được sử dụng trong đề tài

#### Mô hình TF-IDF

1. Giới thiệu mô hình TF-IDF

Mô hình TF-IDF (Term Frequency-Inverse Document Frequency), là một trong những phương pháp truy vấn thông tin cổ điển nhất được sử dụng trong lĩnh vực xử lý ngôn ngữ tự nhiên và truy vấn thông tin.

1. Mô hình TF-IDF trong truy xuất thông tin
2. TF-IDF

* TF(Term frequency) là tần suất xuất hiện của 1 từ w trong văn bản d. Công thức tính: TF(w,d)=len(d)/(w,d)
* IDF( Invert văn bản Frequency) dùng để đánh giá mức độ quan trọng của 1 từ trong văn bản. Khi tính TF mức độ quan trọng của các từ là như nhau. IDF(w, D) = log(M/f(w,D)).
* IDF( Invert Document Frequency) dùng để đánh giá mức độ quan trọng của 1 từ trong văn bản. Khi tính TF mức độ quan trọng của các từ là như nhau. IDF(w, D) = log(M/f(w,D))
* TF-IDF sẽ được tính như sau: TF-IDF(w, d, D) = (len(d)/(w,d)) \* (log(M/f(w,D)))

1. Vector hóa Tài Liệu (văn bản Vectorization):

Bag of Words (BoW):

* Mục tiêu chính của bước vector hóa tài liệu là chuyển đổi các văn bản từ dạng chuỗi thành vector. Phương pháp đơn giản nhất là sử dụng "Bag of Words" (Túi từ).
* Đầu tiên, xây dựng một tập từ vựng cho toàn bộ dataset và sau đó mỗi văn bản được biểu diễn bằng một vector có độ dài bằng kích thước của tập từ vựng. Giá trị tại mỗi vị trí trong vector biểu thị sự xuất hiện (1) hoặc không xuất hiện (0) của từ tương ứng.

TF-IDF Vectorization:

* Ngoài Bag of Words, phương pháp TF-IDF (Term Frequency-Inverse Document Frequency) cũng được sử dụng để vector hóa tài liệu và cải thiện kết quả. Thay vì gán giá trị 1 cho từ xuất hiện, giá trị được tính toán dựa trên tần suất xuất hiện của từ đó trong tài liệu và nghịch đảo tần suất xuất hiện trong toàn bộ dataset.

1. Tính tương đồng (Similarity):

Sau khi tài liệu và câu truy vấn của người dùng được vector hóa, độ tương đồng giữa chúng cần được đo lường. Thường sử dụng độ đo cosine similarity để tính sự tương đồng giữa hai vector.

Độ tương đồng giúp xác định mức độ liên quan giữa câu truy vấn và từng tài liệu trong cơ sở dữ liệu. Càng cao, tài liệu càng có khả năng chứa thông tin phù hợp với câu truy vấn của người dùng.

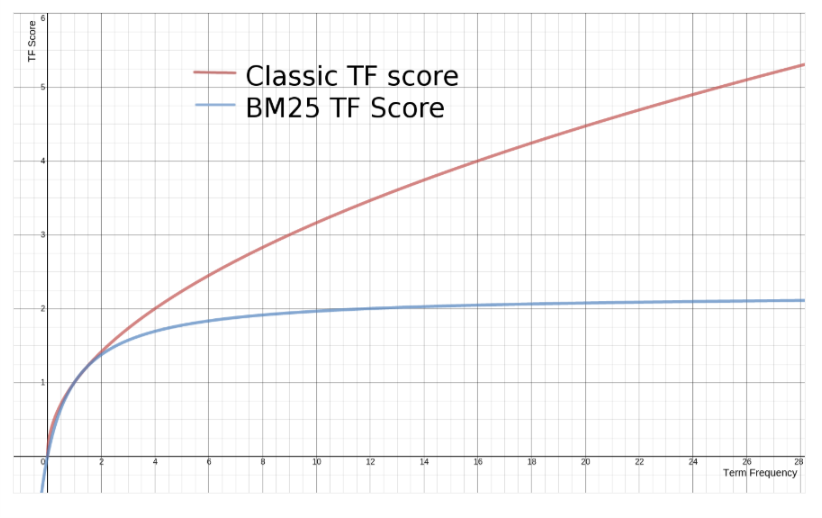
Ví dụ: Trong trường hợp tìm kiếm thông tin về tuyển sinh đại học, nếu người dùng nhập câu truy vấn "điểm chuẩn ngành CNTT tại Đại học Tôn Đức Thắng," hệ thống ODQA sẽ vector hóa câu truy vấn và so sánh với các thông tin về điểm chuẩn của ngành CNTT tại Đại học Tôn Đức Thắng. Các thông tin về điểm chuẩn, ngành học và tên trường có độ tương đồng cao sẽ được hiển thị đầu tiên trong kết quả, giúp người dùng dễ dàng tìm thấy thông tin liên quan đến tuyển sinh ngành học mong muốn.

#### Mô hình BM-25

1. Giới thiệu mô hình BM-25

Mô hình BM-25 là một cải tiến của mô hình TF-IDF, được thiết kế để giải quyết một số hạn chế của mô hình gốc. BM-25 sử dụng cơ chế điều chỉnh tham số để tối ưu hóa hiệu suất truy xuất thông tin. Nó tập trung vào việc xác định sự quan trọng của từ khoá trong tài liệu và câu truy vấn, đồng thời cân nhắc giữa sự xuất hiện của từ khoá và độ dài của tài liệu.

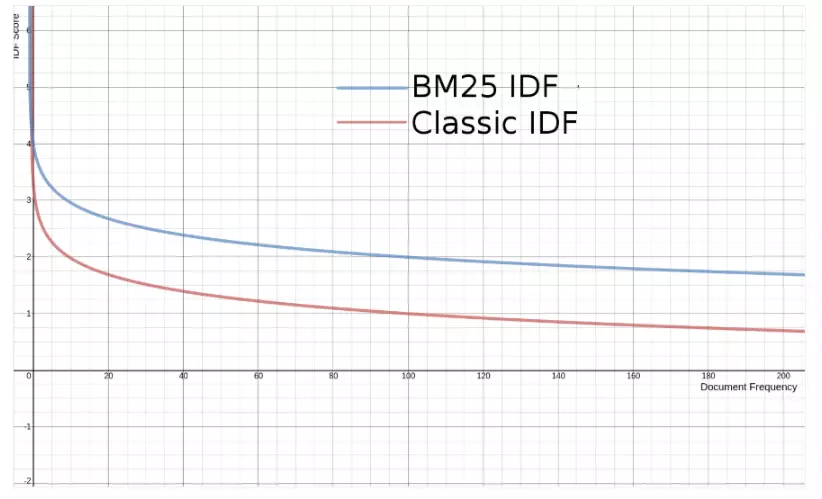
1. Mô hình BM-25 trong truy xuất thông tin
2. TF trong BM-25



Hình 2.5 So sánh điểm giữa TF truyền thống và BM-25

* Trên hình 2.5 , rõ ràng thấy điểm yếu của TF/IDF khiến điểm số từ tần suất xuất hiện (TF) tăng không kiểm soát, tiến gần về vô hạn khi TF tăng lên. Để giảm tác động lớn của TF đối với độ quan trọng (relevance), mô hình BM25 đã điều chỉnh công thức của TF. Điều này dẫn đến việc giới hạn điểm số từ TF tại một giá trị cực đại, và điều này có thể được điều chỉnh linh hoạt theo yêu cầu cụ thể. BM25 không chỉ giúp cải thiện hiệu suất mà còn tăng tính linh hoạt của mô hình đối với độ quan trọng của từ khoá trong tìm kiếm và truy xuất thông tin.
* Công thức tính của TF trong BM-25: ((k + 1) \* tf) / (k + tf). Trong đó, k là hằng số và tf là frequency của term trong văn bản

1. IDF trong BM-25

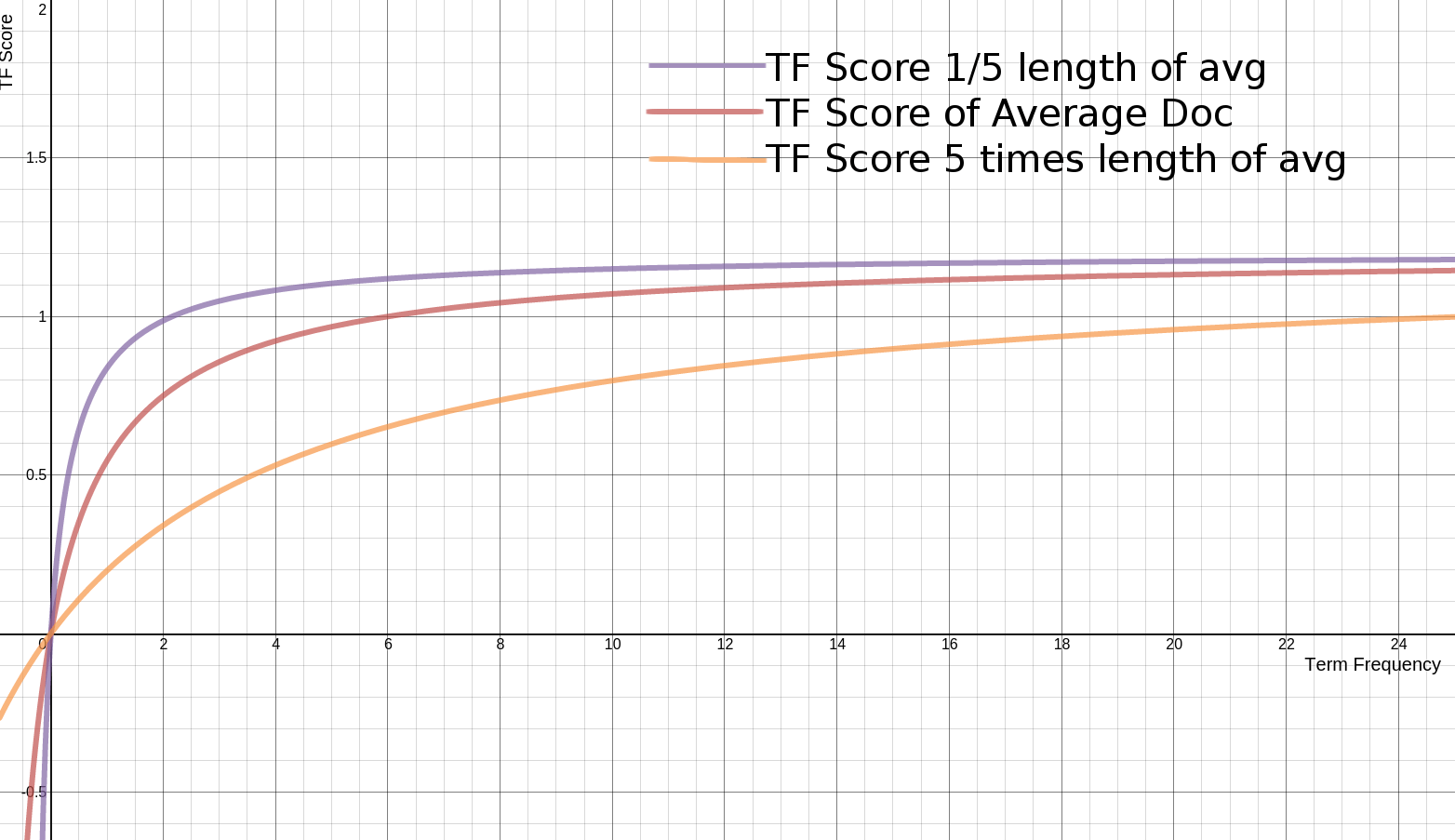


Hình 2.6 So sánh điểm giữa TF truyền thống và BM-25

* Trên hình 2.6, có thể quan sát rằng IDF trong mô hình BM25 có sự tương đồng với IDF trong TF/IDF. Tuy nhiên, BM25 đã thực hiện các điều chỉnh trong công thức tính để tăng khả năng đưa ra điểm số âm khi tần suất xuất hiện của một từ trên toàn bộ index rất cao, tức là khi từ đó xuất hiện rất ít và có tính đặc biệt. Điều này giúp BM25 trở nên linh hoạt hơn đối với các từ hiếm, tránh tình trạng quá ưu ái các từ phổ biến như trong trường hợp IDF của TF/IDF. Sự điều chỉnh này làm tăng khả năng phản ánh độ quan trọng của từ khoá đặc biệt trong việc tìm kiếm và truy xuất thông tin.
* Công thức tính của IDF trong BM-25: IDF(t) = log(1 + (docCount - docFreq + 0.5) / (docFreq + 0.5)). Trong đó, docCount là số lượng văn bản và docFreq là số lượng văn bản chứa term.

1. Dộ dài văn bản trong BM-25

* Công thức cải tiến của TF trong mô hình BM25 thật sự mang lại độ chính xác cao hơn, đặc biệt khi đối mặt với các tình huống có độ dài văn bản đặc biệt ngắn hoặc dài so với độ dài trung bình trong toàn bộ chỉ mục.
* Công thức của TF được điều chỉnh như sau: ((k + 1) \* freq) / (k \* (1.0 - b + b \* L) + freq). Trong đó, b=0.75 (mặc định) và L là tỉ lệ giữa độ dài của văn bản so với độ dài trung bình của tất cả văn bản. L được tính bằng L = fieldLength / avgFieldLength.
* Tùy chỉnh hệ số b trong mô hình của ta có thể ảnh hưởng đáng kể đến cách mà độ dài của văn bản ảnh hưởng đến giá trị TF (Term Frequency). Khi b tiến gần đến 0, tác động của độ dài văn bản trở nên nhỏ hơn, nghĩa là không quan trọng nhiều đối với giá trị TF. Ngược lại, khi b tăng lên, độ dài của văn bản có ảnh hưởng lớn hơn đến giá trị TF.
* Biểu đồ dưới đây minh họa giá trị của TF đối với các độ dài khác nhau của văn bản:



Hình 2.7 So sánh điểm TF khi độ dài văn bản khác nhau

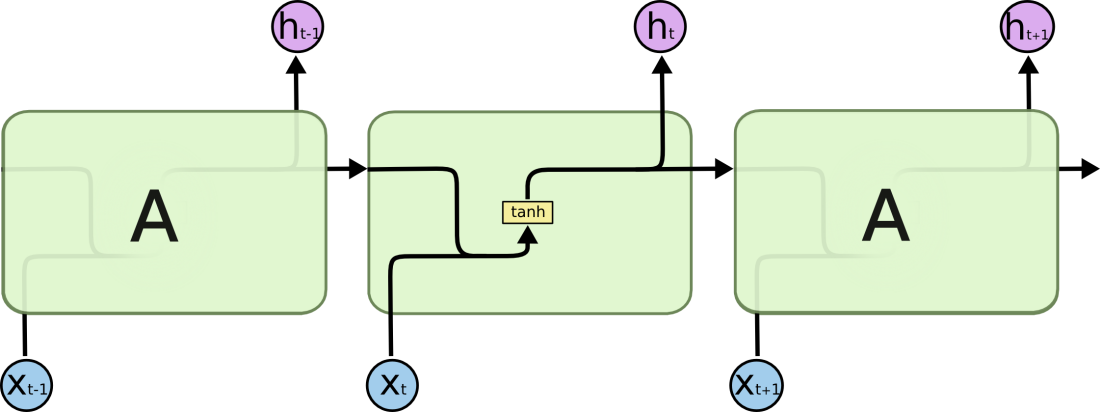
#### Mô hình LSTM

1. Tổng quan mô hình LSTM
2. Khái niệm:

* LSTM, viết tắt của "Long Short-Term Memory," là một kiến trúc đặc biệt trong họ Mạng Nơ-ron Hồi quy (RNN), được giới thiệu bởi Hochreiter & Schmidhuber vào năm 1997. Được biết đến với khả năng học và duy trì sự phụ thuộc dài hạn, LSTM đã trở thành một trong những kiến trúc quan trọng và phổ biến trong lĩnh vực học máy.
* Khác với các RNN thông thường, LSTM đã vượt lên trên những hạn chế của chúng về việc xử lý đạo hàm, làm cho quá trình học trở nên hiệu quả hơn. Tuy nhiên, sự thành công này đến từ sự phức tạp trong cấu trúc của nó, đồng thời vẫn giữ được bản chất của RNN trong việc xử lý dữ liệu theo dạng chuỗi.

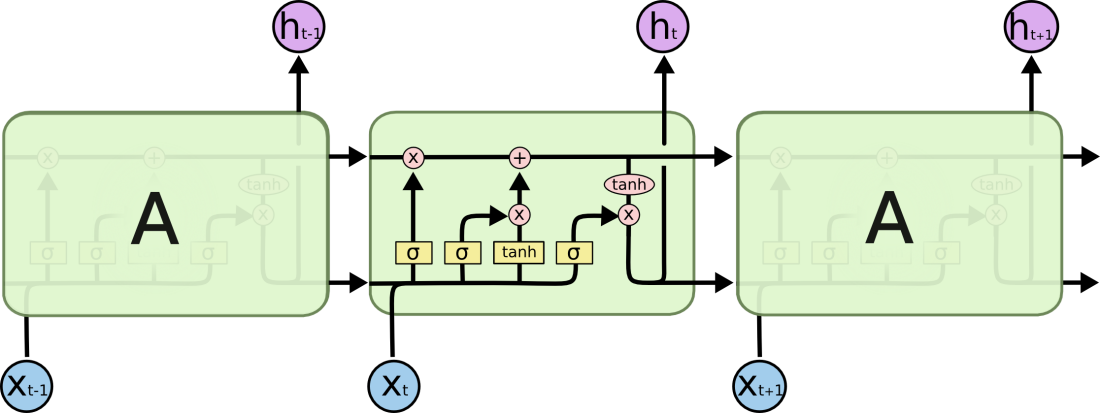
1. Kiến trúc của LSTM

* Trong khi một mạng RNN tiêu chuẩn thường sử dụng một tầng ẩn đơn giản với hàm tanh như hình bên dưới:



Hình 2.8 Sự lặp lại kiến trúc trong mạng RNN chứa một tầng ẩn

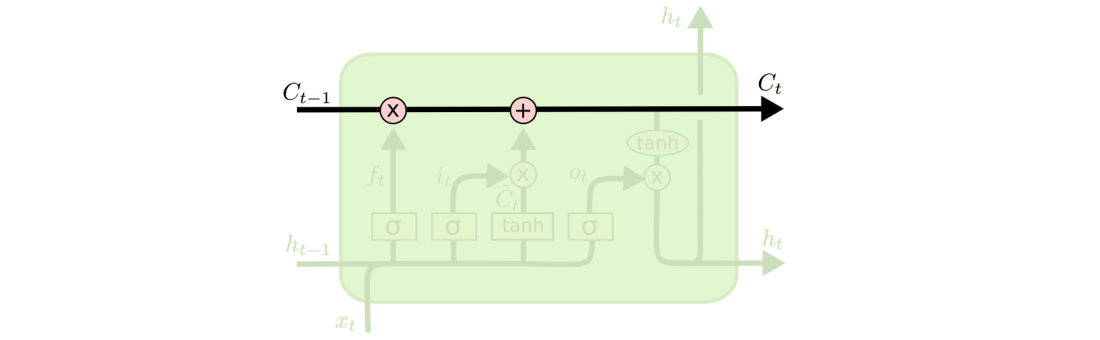
* LSTM không chỉ giống chuỗi kiến trúc của mạng RNN thông thường, mà còn khác biệt với cấu trúc lặp lại phức tạp hơn. Thay vì chỉ có một tầng ẩn, LSTM sử dụng bốn tầng ẩn tương tác - ba sigmoid và một tanh. Sự kết hợp này giúp LSTM hiệu quả trong việc xử lý thông tin dài hạn và đồng thời giải quyết vấn đề đạo hàm mất mát, tạo nên một kiến trúc đặc biệt và mạnh mẽ cho các ứng dụng học máy. Quan sát kiến trúc LSTM ở hình bền dưới:



Hình 2.9 Sự lặp lại kiến trúc trong mạng LSTM chứa 4 tầng ẩn (3 sigmoid và 1 tanh) tương tác

1. Ý tưởng sáng tạo LSTM

* LSTM hoạt động dựa trên ý tưởng chính của việc duy trì một thành phần quan trọng gọi là ô trạng thái (cell state), được mô tả bằng đường chạy ngang qua đỉnh đồ thị. Ô trạng thái này là như một băng chuyền, giúp thông tin chuyển dọc suốt chuỗi mà không bị thay đổi nhiều, nhờ vào các tương tác tuyến tính nhỏ.

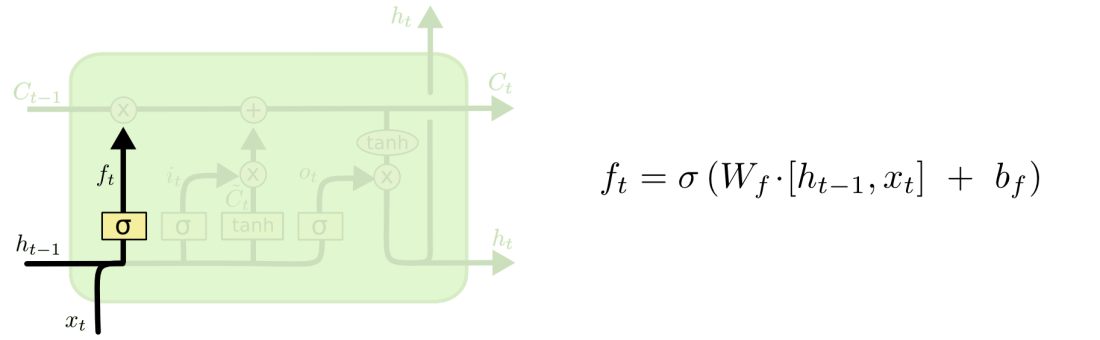


Hình 2.10 Đường đi của ô trạng thái (cell state) trong mạng LSTM

* LSTM có khả năng kiểm soát và điều chỉnh thông tin trong ô trạng thái bằng cách sử dụng các cổng. Các cổng này đóng vai trò quan trọng trong quá trình xóa và thêm thông tin vào ô trạng thái. Mỗi cổng được thiết kế bằng cách sử dụng hàm activation sigmoid, tạo ra một giá trị xác xuất từ 0 đến 1. Giá trị này quyết định mức độ thông tin nào được phép đi qua cổng, với 0 ngụ ý là không cho phép thông tin nào đi qua và 1 là cho phép toàn bộ thông tin đi qua.

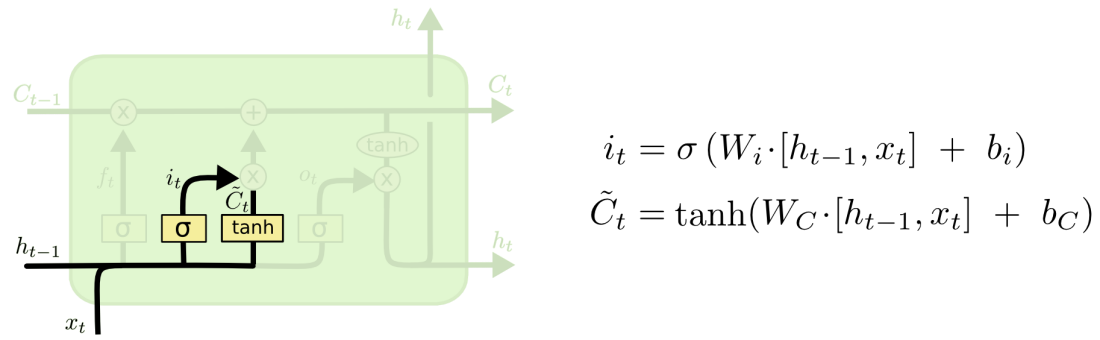
1. Cách hoạt động của LSTM

* Bước đầu tiên trong quá trình hoạt động của LSTM liên quan đến quyết định xem thông tin nào sẽ được chuyển qua ô trạng thái (cell state). Quá trình này được kiểm soát bởi hàm sigmoid trong một tầng được gọi là tầng cổng quên (forget gate layer). Tầng này nhận hai giá trị đầu vào, thường là giá trị trạng thái trước đó ht-1 và đầu vào hiện tại xt, sau đó trả về một giá trị nằm trong khoảng từ 0 đến 1 cho mỗi giá trị trong của ô trạng thái Ct-1. Giá trị bằng 1 thể hiện "giữ toàn bộ thông tin," trong khi giá trị bằng 0 thể hiện "bỏ qua toàn bộ thông tin."



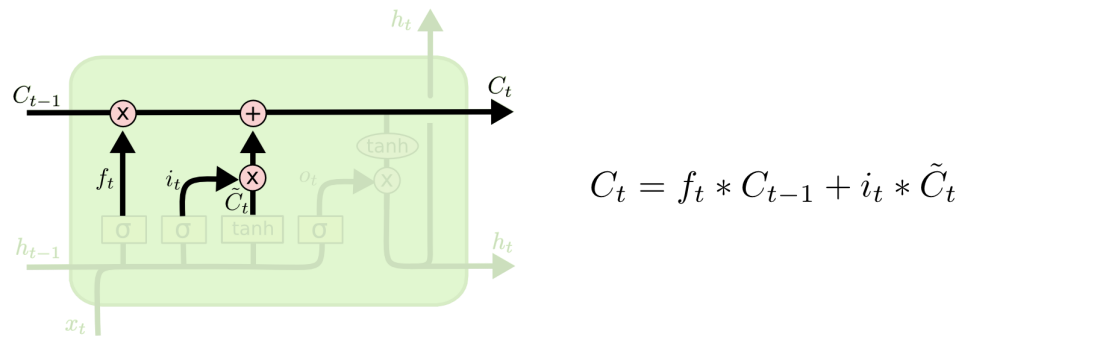
Hình 2.11 Tầng cổng quên (forget layer)

* Bước tiếp theo trong quá trình của LSTM liên quan đến quyết định loại thông tin nào sẽ được lưu trữ trong ô trạng thái. Quá trình này được thực hiện thông qua hai phần chính. Phần đầu tiên bao gồm một tầng ẩn của hàm sigmoid được gọi là tầng cổng vào (input gate layer). Tầng này quyết định giá trị nào sẽ được cập nhật trong ô trạng thái. Đầu vào cho tầng cổng này thường là kết hợp của thông tin trạng thái trước đó và thông tin đầu vào hiện tại. Kết quả của tầng cổng vào này là một giá trị xác xuất từ 0 đến 1, biểu thị mức độ thông tin nào sẽ được cập nhật. Phần thứ hai của bước này là tầng ẩn của hàm tanh, tạo ra một véc tơ giá trị trạng thái mới. Điều này giúp xác định thông tin cần được thêm vào trạng thái mới. Cuối cùng, kết quả của cả hai tầng (cổng vào và hàm tanh) được kết hợp để tạo ra một cập nhật cho trạng thái hiện tại Ct, làm giàu thông tin và chuẩn bị cho các bước tiếp theo trong quá trình học của mô hình LSTM.



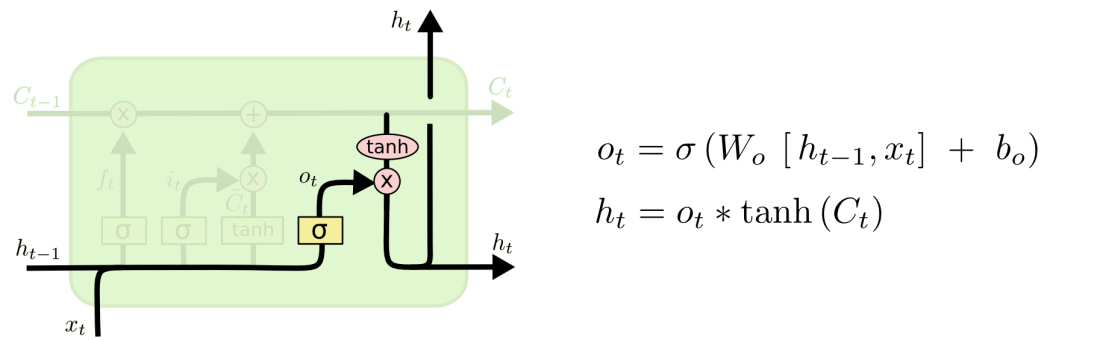
Hình 2.12 Cập nhật giá trị cho ô trạng thái bằng cách kết hợp 2 kết quả từ tầng cổng vào và tẩng ẩn hàm tanh

* Ở bước này, chúng ta tiến hành cập nhật ô trạng thái cũ Ct-1 sang trạng thái mới Ct. Những quyết định đã được đưa ra ở các bước trước đó chỉ cần được thực hiện ở đây. Đầu tiên, chúng ta nhân trạng thái cũ với giá trị đầu ra của tầng cổng quên. Điều này có tác dụng quên đi những thông tin cần được loại bỏ theo xác suất quyết định tại bước quên. Tiếp theo, chúng ta cộng thêm phần tử đề cử, một giá trị mới được tính toán tương ứng với lượng thông tin mới được cập nhật vào mỗi giá trị trạng thái. Bước này giúp làm giàu trạng thái mới với thông tin quan trọng và cần thiết cho dự đoán tiếp theo của mô hình LSTM.



Hình 2.13 Ô trạng thái mới

* Cuối cùng, để quyết định đầu ra của mô hình LSTM, chúng ta sử dụng ô trạng thái được làm mới, nhưng được lọc qua một quá trình điều chỉnh. Quá trình này bao gồm hai bước chính. Đầu tiên, chúng ta chạy ô trạng thái mới qua một tầng sigmoid, nơi mà quyết định phần nào của ô trạng thái sẽ được đưa ra ở đầu ra. Điều này tương ứng với việc lọc thông tin và chọn ra những phần quan trọng. Tiếp theo, ô trạng thái mới được đưa qua hàm tanh để đảm bảo giá trị nằm trong khoảng từ -1 đến 1. Sau đó, kết quả của nó được nhân với đầu ra của một cổng sigmoid khác. Bước này giúp chỉ giữ lại phần của ô trạng thái mà chúng ta quyết định là quan trọng, tạo ra đầu ra cuối cùng của mô hình LSTM. Điều này giúp mô hình tạo ra đầu ra dự đoán dựa trên thông tin quan trọng được học và giữ lại từ quá trình học máy trước đó.

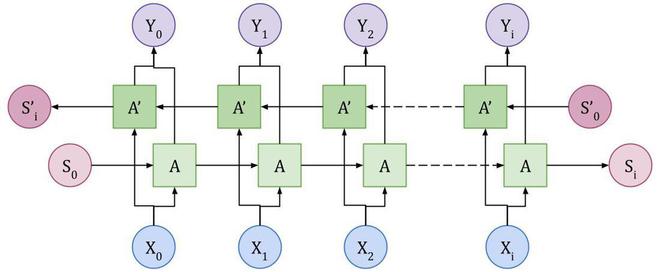


Hình 2.14 Điều chỉnh thông tin ở đầu ra thông qua hàm tanh

1. Các biến thể của LSTM

* GRU (Gated Recurrent Unit): GRU là một biến thể của LSTM được thiết kế để giảm độ phức tạp của LSTM, giữ nguyên khả năng xử lý dài hạn. Nó sử dụng cổng quên và cổng đầu vào, nhưng chỉ có một tầng ẩn, giảm thiểu lượng thông tin cần lưu trữ
* Peephole LSTM: Biến thể này mở rộng LSTM bằng cách cho phép các cổng (quên, đầu vào, đầu ra) trực tiếp xem thông tin từ ô trạng thái. Điều này giúp mô hình có thêm thông tin context khi quyết định cập nhật.
* Depth Gated LSTM: Được thiết kế để tăng hiệu suất của mô hình, Depth Gated LSTM sử dụng nhiều tầng ẩn thay vì chỉ một tầng như trong LSTM truyền thống.
* LSTM with Attention Mechanism: Kết hợp LSTM với cơ chế chú ý giúp mô hình tập trung vào các phần quan trọng của đầu vào, cải thiện khả năng hiểu ngữ cảnh trong các tác vụ như dịch máy hoặc mô hình ngôn ngữ tự nhiên.
* Bi-directional LSTM (BiLSTM): Mô hình này sử dụng hai hướng, chuyển thuận và ngược, để xem xét cả quá khứ và tương lai của mỗi ngữ cảnh, giúp cải thiện khả năng dự đoán trong các chuỗi dữ liệu.

1. BiLSTM
2. Khái niệm: BiLSTM là một kiến trúc mô hình học máy, là biến thể của mạng LSTM được thiết kế để hiểu biểu diễn của dữ liệu chuỗi từ cả quá khứ và tương lai. Sự đặc biệt của BiLSTM nằm ở khả năng xử lý ngữ cảnh hai chiều, làm cho nó trở thành lựa chọn mạnh mẽ cho các nhiệm vụ yêu cầu sự hiểu biết sâu sắc về chuỗi dữ liệu.
3. Kiến trúc của BiLSTM: BiLSTM bao gồm hai phần chính: một LSTM thuận (chuyển) và một LSTM ngược (ngược). Mỗi bộ LSTM này thực hiện việc xử lý dữ liệu theo từng chiều (trong và ngược lại) đồng thời. Các ô trạng thái và cổng của LSTM đảm bảo rằng thông tin từ cả hai chiều được tích hợp một cách linh hoạt. Ảnh bên dưới là kiến trúc của mô hình BiLSTM:



Hình 2.15 Kiến trúc của mô hình BiLSTM

* Quy trình làm việc của BiLSTM như sau:
  + Hướng Chuyển (Thuận): Dữ liệu được đưa qua LSTM theo chiều chuyển, từ trái sang phải. Các ô trạng thái và cổng của LSTM thuận cập nhật thông tin từ quá khứ.
  + Hướng Ngược (Ngược): Dữ liệu được đưa qua LSTM theo chiều ngược, từ phải sang trái. Các ô trạng thái và cổng của LSTM ngược cập nhật thông tin từ tương lai.
  + Kết Hợp: Ô trạng thái cuối cùng là tổ hợp của ô trạng thái cuối cùng từ LSTM thuận và LSTM ngược. Đầu ra cuối cùng là kết hợp của đầu ra từ cả hai hướng.

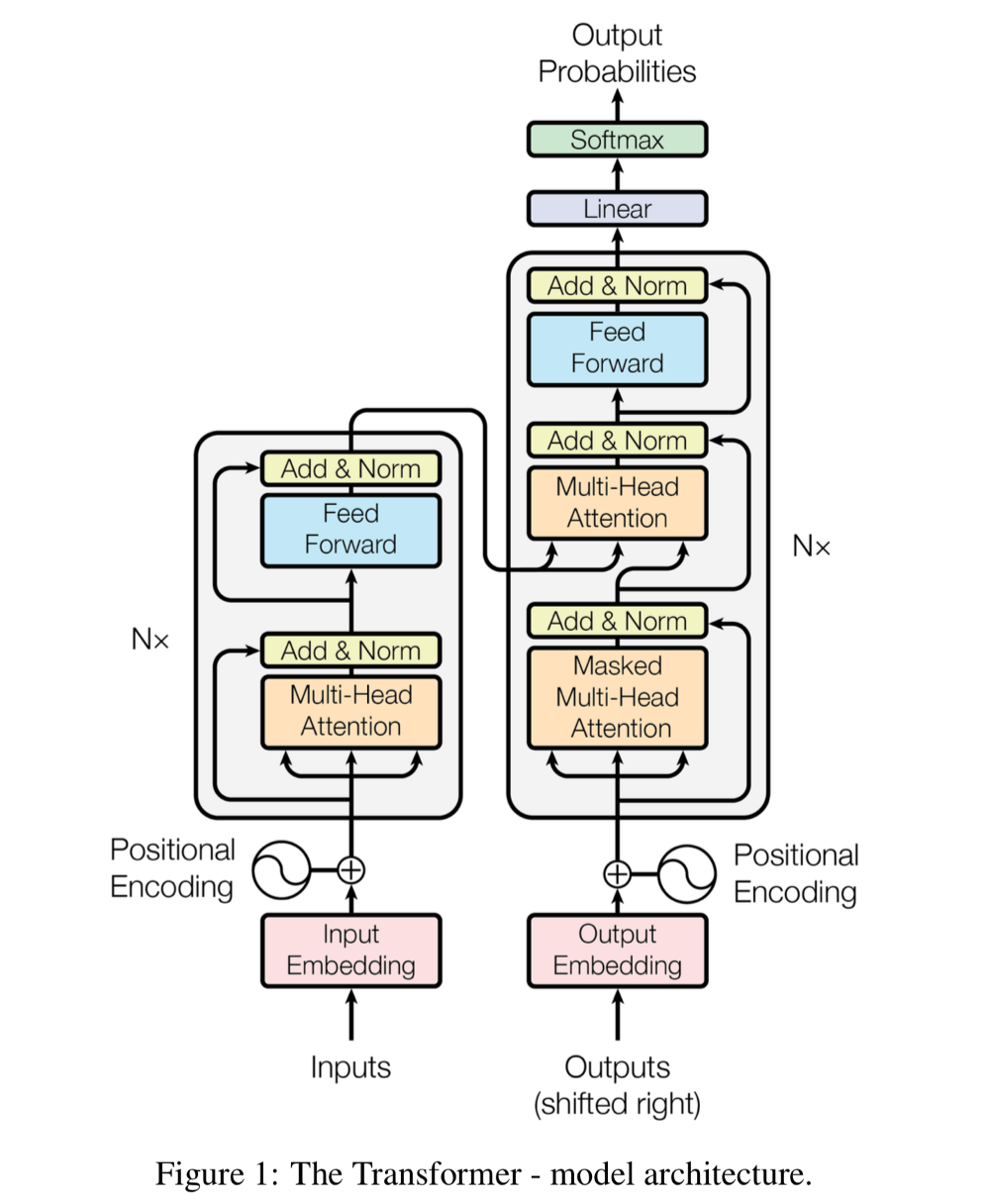
1. Ứng dụng

* Xử lý ngôn ngữ tự nhiên (NLP): Trong nhiều tác vụ NLP như dự đoán từ tiếp theo, phân loại câu, hay dịch máy, BiLSTM có thể hiệu quả bởi khả năng nắm bắt ngữ cảnh toàn diện.
* Nhận dạng giọng nói: BiLSTM có thể được sử dụng để nhận dạng và hiểu giọng nói với khả năng xử lý thông tin từ cả quá khứ và tương lai của âm thanh.
* Dự đoán chuỗi thời gian: Trong dự đoán chuỗi thời gian, BiLSTM có thể giúp mô hình hiểu các mối quan hệ phức tạp giữa các điểm dữ liệu liên tục.

#### Mô hình BERT

1. Tổng quan mô hình BERT
2. Khái niệm: BERT, viết tắt của "Bidirectional Encoder Representations from Transformers," là một mô hình ngôn ngữ tiên tiến trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP). Nó được giới thiệu bởi Google Research vào năm 2018 và đã đạt được những thành công ấn tượng trong nhiều ứng dụng NLP.
3. Encoder và Decoder trong BERT

* Mô hình seq2seq mà chúng ta sẽ đề cập có kiến trúc đặc biệt, không sử dụng các lớp Recurrent Neural Network (RNN) mà thay vào đó sử dụng các lớp attention để nhúng thông tin từ các từ trong câu. Mô hình này bao gồm hai phần chính: encoder và decoder.

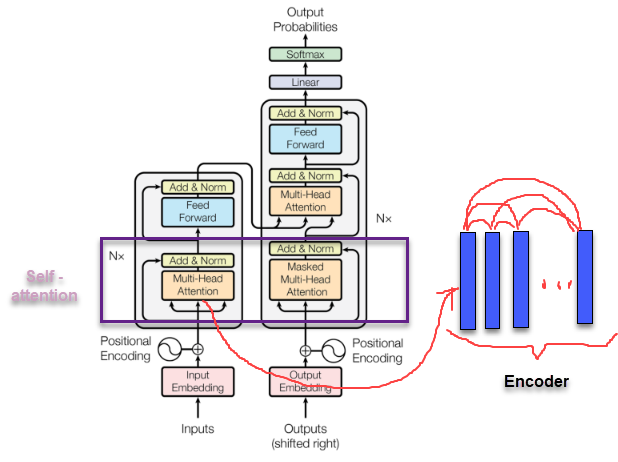


Hình 2.16 Kiến trúc mô hình transformer kết hợp attention

* Encoder: Sử dụng lớp attention để nhúng thông tin từ các từ trong câu nguồn. Mỗi từ được biểu diễn dưới dạng vector và đưa vào lớp attention để tạo ra một biểu diễn tổng hợp cho toàn bộ câu. Các biểu diễn này không sử dụng RNN mà chủ yếu tập trung vào việc tạo ra trọng số quan trọng cho mỗi từ thông qua cơ chế attention.
* Decoder: Cũng sử dụng lớp attention để nhúng thông tin từ encoder. Khác biệt ở chỗ, decoder sử dụng thông tin đã được nhúng từ encoder để tạo ra một chuỗi đầu ra. Mỗi từ trong chuỗi đầu ra được sinh ra dựa trên thông tin từ cả chuỗi đầu ra trước đó và encoder, giúp mô hình hiểu được bối cảnh của câu.

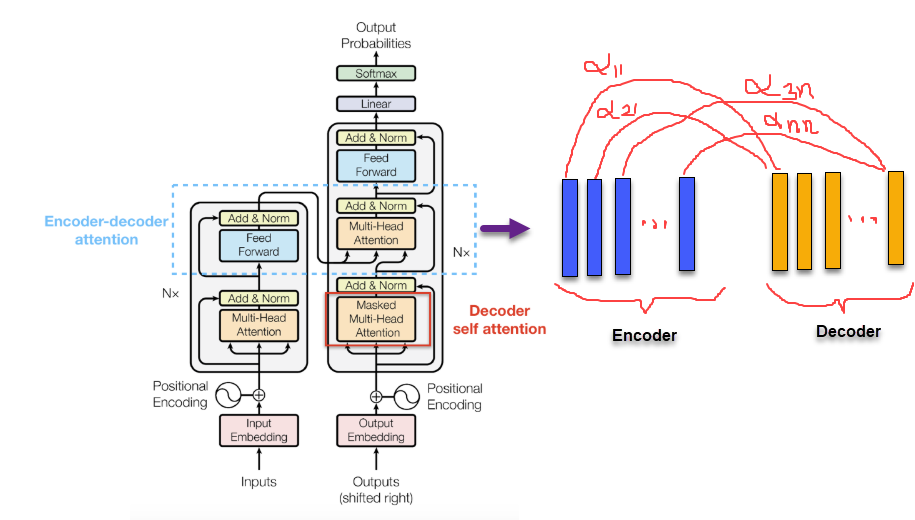
1. Self-attention và encoder-decoder attention

* Mô hình Transformer thực sự độc đáo với việc sử dụng hai dạng attention khác nhau: self-attention và encoder-decoder attention, giúp cải thiện khả năng hiểu bối cảnh của câu trong quá trình huấn luyện.
* Self-attention: Áp dụng trong cùng một câu đầu vào (sequence), có thể ở pha encoder hoặc pha decoder. Được thực hiện bởi các lớp Multi-Head Attention ở đầu vào của cả hai pha. Cho phép mô hình tập trung vào các phần tử quan trọng khác nhau của câu đầu vào, tăng khả năng hiểu cấu trúc và ý nghĩa của câu.



Hình 2.17 Sơ đồ vị trí áp dụng self-attention trong kiến trúc transformer

* Encoder-decoder attention: Được sử dụng để tạo ra véc tơ context từ encoder mà có ý nghĩa trong ngữ cảnh của decoder. Điều này giúp mô hình "nhìn" vào các phần quan trọng của câu nguồn khi tạo ra câu đích. Véc tơ context được tính dựa trên sự tương quan giữa véc tơ decoder và các véc tơ embedding của encoder. Sau đó, thông tin này được sử dụng để dự đoán từ tiếp theo trong chuỗi đầu ra.



Hình 2.18 Vị trí áp dụng encoder-decoder attention

1. Kiến trúc

* Các biến thay đổi chính giữa các phiên bản của mô hình BERT là tập trung vào ba tham số quan trọng của kiến trúc Transformer:
  + Số lượng block sub-layers trong Transformer (L) : Đây là số lượng các block chứa các sub-layers trong mô hình Transformer. Sự thay đổi này ảnh hưởng đến khả năng học và biểu diễn của mô hình.
  + Kích thước của embedding vector (hidden size) (H): Là kích thước của vector embedding cho mỗi từ trong câu. Sự tăng kích thước này có thể cải thiện khả năng biểu diễn của mô hình đối với thông tin phức tạp hơn.
  + Số lượng head trong multi-head layer (A): Đây là số lượng các head trong lớp multi-head attention, mỗi head thực hiện một self-attention. Sự thay đổi này có thể cải thiện khả năng mô hình tập trung vào các mối quan hệ khác nhau trong câu.
* Tên gọi của 2 kiến trúc này là:
  + BERTBASE (L = 12, H = 768, A = 12) Tổng 110 triệu tham số
  + BERTLARGE (L = 24, H = 768, A = 16) Tổng 340 triệu tham số
* Như vậy ở kiến trúc BERTLARGE chúng ta tăng gấp đôi số layer, tăng kích thước hidden size của embedding véc tơ gấp 1.33 lần và tăng số lượng head trong multi-head layer gấp 1.33 lần.

1. Fine-tuning

* Một đặc điểm độc đáo của BERT mà các mô hình nhúng từ trước đây chưa có là khả năng fine-tuning của kết quả huấn luyện. Chúng ta có khả năng bổ sung một tầng đầu ra vào kiến trúc mô hình để tinh chỉnh linh hoạt theo yêu cầu của nhiệm vụ huấn luyện cụ thể.

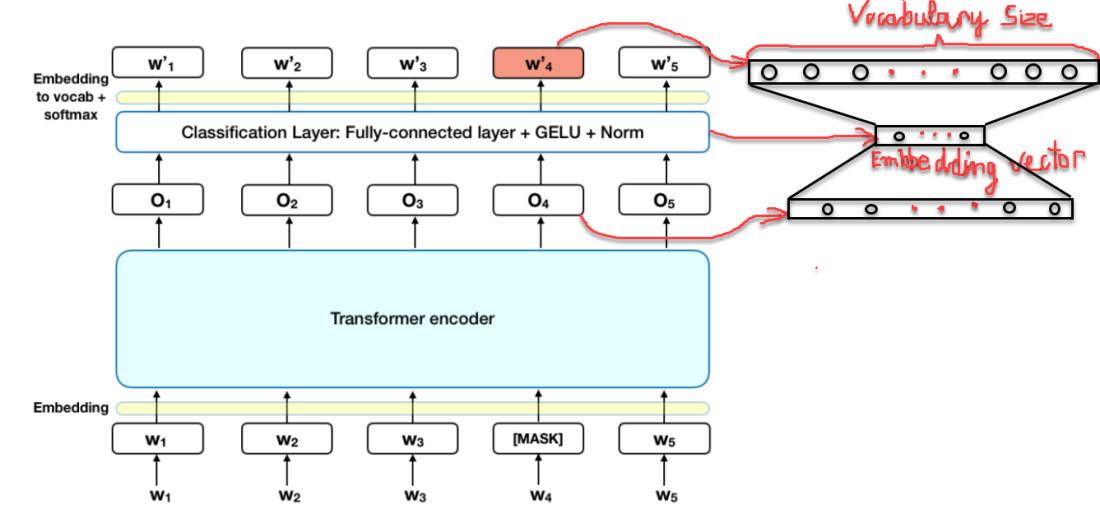


Hình 2.19 Toàn bộ tiến trình pre-training và fine-tuning của BERT

* Một kiến trúc đồng nhất được áp dụng cả cho mô hình pre-training và mô hình fine-tuning. Chúng ta sử dụng cùng một bộ tham số pretrain để khởi tạo mô hình cho các nhiệm vụ down stream khác nhau. Trong suốt quá trình fine-tuning, tất cả các tham số của các lớp học chuyển giao sẽ được điều chỉnh linh hoạt. Nếu ta xem xét các tác vụ đòi hỏi đầu vào là một cặp chuỗi (pair-sequence) như câu hỏi và câu trả lời, chúng ta sẽ mở rộng kiến trúc bằng cách thêm vào đầu câu token đặc biệt là [CLS], và ở giữa cặp câu là token [SEP] để đánh dấu sự phân chia giữa hai đoạn văn bản. Điều này giúp mô hình hiểu rõ cấu trúc của cặp câu và cũng hữu ích cho các tác vụ như hỏi đáp (question answering) khi cần phân biệt giữa ngữ cảnh và câu trả lời.
* Quá trình áp dụng fine-tuning sẽ như sau:
  + Bước 1: Mọi token trong cặp câu sẽ được nhúng bằng các vector từ pretrain model. Cả hai token đặc biệt là [CLS] và [SEP] cũng được bao gồm, để đánh dấu vị trí bắt đầu của câu hỏi và vị trí ngăn cách giữa hai câu. Các token này sẽ tham gia dự báo ở đầu ra để xác định các vị trí Start/End Span của câu trả lời.
  + Bước 2: Các vector nhúng sau đó sẽ trải qua một kiến trúc multi-head attention với nhiều khối mã (thường là 6, 12 hoặc 24 khối tùy thuộc vào kiến trúc BERT). Kết quả là một vector đầu ra từ bộ mã hóa.
  + Bước 3: Để dự đoán phân phối xác suất cho mỗi vị trí từ ở bộ giải mã, tại mỗi bước thời gian, ta truyền vào bộ giải mã vector đầu ra từ bộ mã hóa và vector nhúng đầu vào từ bộ giải mã để tính attention giữa bộ mã hóa và bộ giải mã (chi tiết về attention giữa encoder-decoder có thể được tìm hiểu thêm ở mục 2.1.1). Sau đó, thông qua một lớp tuyến tính và hàm softmax, ta thu được phân phối xác suất cho đầu ra tại từng bước thời gian.
  + Bước 4: Kết quả đầu ra từ transformer sẽ được so sánh với câu hỏi ở đầu vào để cố định kết quả của câu hỏi. Các vị trí còn lại sẽ tương ứng với các thành phần mở rộng Start/End Span tương ứng với câu trả lời được tìm thấy trong cặp câu.
  + Lưu ý rằng trong quá trình huấn luyện, tất cả các tham số của mô hình BERT (sau khi loại bỏ lớp tuyến tính đỉnh) sẽ được fine-tune. Chúng ta sẽ tiếp tục huấn luyện lại từ đầu các tham số của lớp tuyến tính mà chúng ta thêm vào kiến trúc BERT để tùy chỉnh phù hợp với nhiệm vụ cụ thể.

1. Masked ML (MLM)

* Sơ đồ huấn luyện BERT theo tác vụ Masked ML là quá trình đào tạo mô hình ngôn ngữ không giám sát (unsupervised) trên các bộ dữ liệu lớn, trong đó một phần của văn bản đã bị ẩn đi (masked) và mô hình cố gắng dự đoán nội dung của những vị trí bị ẩn đó. Dưới đây là hình ảnh mô tả sơ đồ này:



Hình 2.20 Sơ đồ kiến trúc BERT cho tác vụ Masked ML

* Tác vụ Masked ML trong kiến trúc BERT đặt ra mục tiêu ẩn khoảng 15% token của câu đầu vào bằng [MASK], với ước lượng này giúp mô hình dự đoán giá trị gốc của các từ bị che dấu. Trong quá trình này, mô hình dựa vào các từ không bị che dấu xung quanh [MASK] để hiểu bối cảnh và dự báo từ được che dấu. Tỷ lệ 15% được chọn để đảm bảo sự đa dạng trong dữ liệu, với 85% còn lại là bối cảnh đóng vai trò quan trọng.
* Kiến trúc của BERT vẫn giữ nguyên tính chất seq2seq với hai pha encoder và decoder. Tuy nhiên, trong tác vụ Masked ML, chỉ pha encoder được sử dụng để học biểu diễn từ. Sau khi thực hiện self-attention và feed forward trong Transformer encoder, mô hình thu được các véc tơ embedding ở đầu ra.
* Để tính toán phân phối xác suất cho từ đầu ra, một fully connected layer được thêm vào ngay sau Transformer Encoder, và hàm softmax được áp dụng. Số lượng units của fully connected layer phải bằng với kích thước của từ điển.
* Véc tơ embedding của từ bị che dấu tại vị trí MASK thu được bằng cách giảm chiều của véc tơ sau khi đi qua fully connected layer. Hàm loss function của BERT chỉ tính mất mát từ những từ bị che dấu, điều này tăng cường ý thức về bối cảnh và chất lượng biểu diễn.

1. Next Sentence Prediction (NSP)

* Đây là một bài toán phân loại học có giám sát với 2 nhãn, hay còn gọi là phân loại nhị phân. Mô hình nhận đầu vào là một cặp câu (pair-sequence), trong đó 50% câu thứ hai được lựa chọn là câu tiếp theo của câu thứ nhất và 50% được lựa chọn một cách ngẫu nhiên từ bộ văn bản mà không có mối liên hệ gì với câu thứ nhất. Nhãn của mô hình được gán là IsNext nếu cặp câu là liên tiếp, và NotNext nếu cặp câu không liên tiếp.
* Tương tự như mô hình Question and Answering, các vị trí đầu câu thứ nhất được đánh dấu bằng token [CLS], và vị trí cuối cùng của mỗi câu được đánh dấu bằng token [SEP]. Những token này có tác dụng nhận biết các vị trí bắt đầu và kết thúc của từng câu, tạo ra cặp câu có cấu trúc đặc biệt để mô hình có thể học và dự đoán mối quan hệ giữa chúng.

1. PhoBERT
2. Khái niệm: PhoBERT là viết tắt của "Pho" (tiếng Việt gọi chung cho phở) và "BERT," chỉ rõ sự kế thừa từ kiến trúc BERT. Mô hình này có khả năng hiểu biểu diễn ngôn ngữ tự nhiên tiếng Việt một cách hiệu quả thông qua việc học biểu diễn từ ngữ theo cả hai hướng, từ trái sang phải và từ phải sang trái.
3. Kiến trúc: Tương tự như BERT.
4. Các đặc điểm chính của PhoBERT:

* PhoBERT được huấn luyện trước (pre-trained) với mô hình ngôn ngữ dựa trên tiếng Việt mà không chú trọng đến nhiều ngôn ngữ khác.
* PhoBERT có hai phiên bản chính là PhoBERTBASE và PhoBERTLARGE. Phiên bản base bao gồm 12 blocks Transformer, trong khi phiên bản large có 24 blocks Transformer.
* PhoBERT được huấn luyện trên khoảng 20GB dữ liệu tiếng Việt, bao gồm khoảng 1GB từ Vietnamese Wikipedia corpus và 19GB còn lại từ Vietnamese news corpus. Lượng dữ liệu này đủ lớn để đảm bảo mô hình có khả năng tổng quát hóa và hiểu sâu về ngôn ngữ tiếng Việt.
* Để chuẩn bị dữ liệu đầu vào, PhoBERT sử dụng RDRSegmenter của VnCoreNLP để tách từ trước khi trải qua quá trình mã hóa BPE (Byte Pair Encoding) encoder.
* PhoBERT, theo lối tiếp cận của RoBERTa, chỉ sử dụng nhiệm vụ Masked Language Model (MLM) trong quá trình huấn luyện. Nhiệm vụ này làm cho mô hình học biểu diễn ngôn ngữ sâu sắc thông qua việc dự đoán từ bị ẩn trong văn bản.

#### Mô hình GPT

1. Tổng quan mô hình GPT
2. Khái niệm:
3. Unsupervised pre-training
4. Supervised fine-tuning
5. Task-specific input transformations
6. GPT-2

#### Mô hình XLM-RoBERTa

1. Tổng quan mô hình XLM-RoBERTa
2. Khái niệm: XLM-RoBERTa là một mô hình ngôn ngữ tiếng Anh được xây dựng trên cơ sở của RoBERTa , một biến thể của mô hình BERT được tinh chỉnh để cải thiện hiệu suất trong việc hiểu ngôn ngữ tự nhiên. Mô hình này được phát triển bởi Facebook AI Research (FAIR).
3. Kiến trúc: Tương tự BERT
4. Đặc điểm cải tiến so với BERT

* Đa ngôn ngữ: Trong khi BERT tập trung chủ yếu vào tiếng Anh, XLM-RoBERTa được tinh chỉnh để hiểu và làm việc trên nhiều ngôn ngữ. Điều này giúp mô hình tổng quát hóa tốt hơn trên các ngôn ngữ khác nhau và có khả năng chuyển giao tốt hơn giữa các ngôn ngữ.
* Tăng cường đào tạo: XLM-RoBERTa được đào tạo trên một lượng lớn dữ liệu từ nhiều nguồn ngữ cảnh khác nhau trên Internet, giúp nó hiểu được đa dạng và phong phú của ngôn ngữ tự nhiên. Quá trình này giúp mô hình tổng quát hóa tốt hơn và cải thiện khả năng hiểu và sinh ra ngôn ngữ.
* Kiến trúc mô hình: Mặc dù XLM-RoBERTa và BERT đều sử dụng kiến trúc mã hóa dựa trên Transformer, XLM-RoBERTa có thể có một số cải tiến trong việc tối ưu hóa kiến trúc mạng, hoặc sử dụng các phương pháp đào tạo khác nhau để cải thiện hiệu suất và tổng quát hóa.
* Hiệu suất đa ngôn ngữ: XLM-RoBERTa có khả năng tổng quát hóa tốt hơn trên các ngôn ngữ khác nhau, có thể giảm được sự chênh lệch hiệu suất giữa các ngôn ngữ so với BERT.

1. MRCQuestionAnswering
2. Khái niệm: MRCQuestionAnswering là một dạng của MRC, nơi mà hệ thống máy tính được huấn luyện để trả lời các câu hỏi dựa trên nội dung của một hoặc nhiều tài liệu.
3. Các đặc điểm chính MRCQuestionAnswering

* Nền tảng mô hình: Xây dựng trên nền tảng của XLM-RoBERTa, một biến thể của mô hình RoBERTa được tinh chỉnh để làm nhiệm vụ trả lời câu hỏi và trích xuất thông tin từ văn bản.
* Fine-tuning: Mô hình đã được fine-tuning cho mục đích cụ thể là trả lời câu hỏi và trích xuất thông tin từ văn bản. Việc này giúp mô hình học được cách tốt nhất để thực hiện nhiệm vụ này dựa trên dữ liệu huấn luyện.
* Ngôn ngữ chính: Ngôn ngữ chính của mô hình là tiếng Việt, nhưng cũng có khả năng hoạt động tốt với tiếng Anh.
* Dữ liệu huấn luyện: Sử dụng các tập dữ liệu đa dạng như Squad 2.0, mailong25, UIT-ViQuAD và MultiLingual Question Answering để huấn luyện mô hình. Điều này giúp mô hình học được từ nhiều nguồn dữ liệu và làm quen với nhiều loại văn bản khác nhau.

## Module Website

### MongoDB và Mongoose

#### Tổng quan về MongoDB:

MongoDB là một database hướng tài liệu (document) hay còn được gọi một dạng NoSQL database. Vì thế, MongoDB sẽ tránh cấu trúc table-based của relational database để thích ứng với các tài liệu như JSON có một schema rất linh hoạt gọi là BSON. MongoDB sử dụng lưu trữ dữ liệu dưới dạng Document JSON nên mỗi một collection sẽ các các kích cỡ và các document khác nhau. Các dữ liệu được lưu trữ trong document kiểu JSON nên tốc độ truy vấn sẽ rất nhanh.

#### Ưu và nhược điểm của MongoDB:

1. Ưu điểm

* Dữ liệu lữu trữ phi cấu trúc, không có tính ràng buộc, hiệu suất lớn, dễ dàng mở rộng và lưu trữ.
* Dữ liệu được ghi đệm (cache) lên RAM, hạn chế truy cập vào phần cứng nên tốc độ đọc và ghi cao.

1. Nhược điểm

* Không được sử dụng cho các mô hình giao dịch cho các giao dịch yêu cầu độ chính xác cao do không có tính ràng buộc.
* Dữ liệu lấy RAM làm trọng tâm hoạt động vì vậy hoạt động yêu cầu một bộ nhớ RAM lớn.
* Khả năng mất dữ liệu khá cao do mọi thay đổi về dữ liệu mặc định đều chưa được ghi xuống ở cứng ngay lập tức.

#### Tổng quan về Mongoose:

Mongoose là một thư viện mô hình hóa đối tượng (Object Data Model - ODM) cho MongoDB và Node.js. Nó quản lý mối quan hệ giữa dữ liệu, cung cấp sự xác nhận giản đồ và được sử dụng để dịch giữa các đối tượng trong mã và biểu diễn các đối tượng trong MongoDB.

#### Ưu và nhược điểm của Mongoose:

1. Ưu điểm

* Mongoose đã tích hợp khả năng xác thực dữ liệu, cho phép kiểm soát dữ liệu được thêm vào hoặc cập nhật vào database.
* Tiền định nghĩa model cho mỗi collection.
* Cho phép áp dụng tính ràng buộc lên mỗi document trong collection.

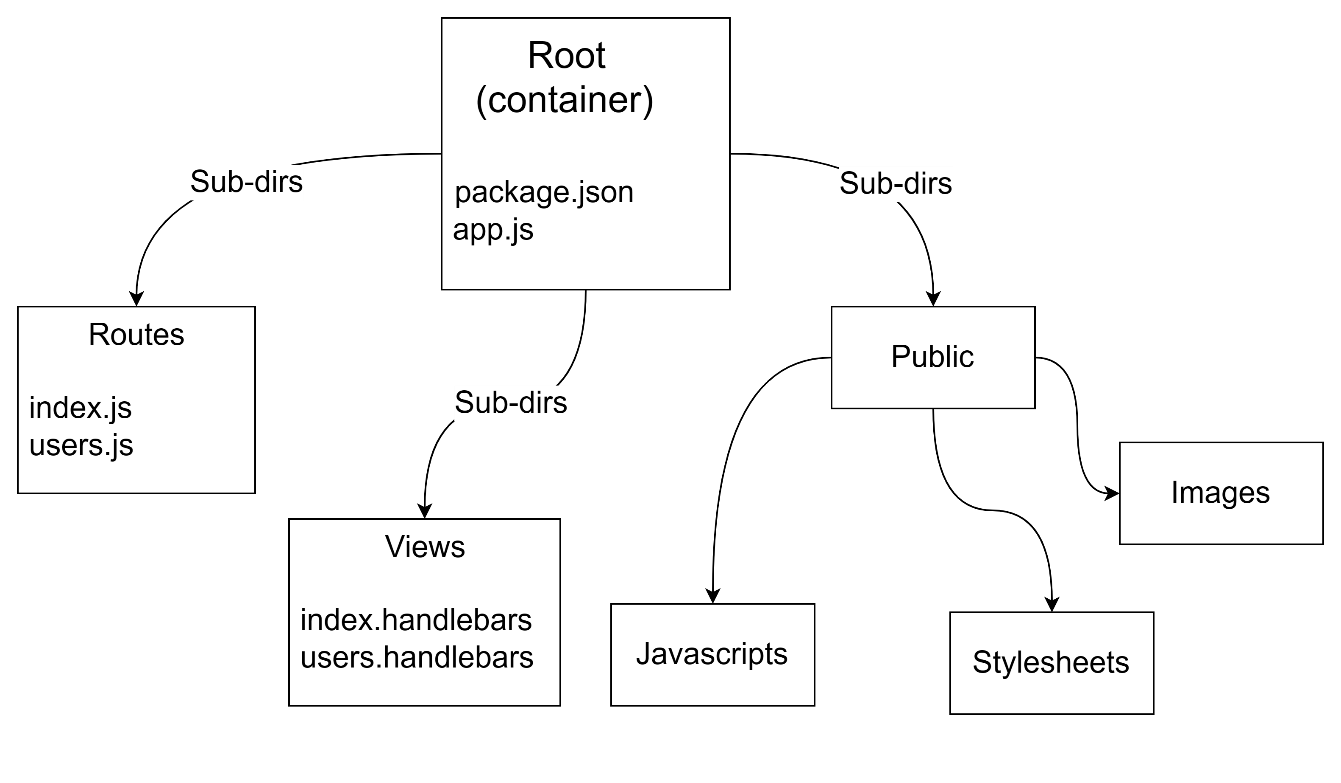
1. Nhược điểm

* Chậm hơn so với việc sử dụng trực tiếp MongoDB.
* Mongoose thêm một lớp trừu tượng (abstraction layer), điều này làm tăng độ phức tạp (complexity) của ứng dụng.

### ExpressJS

#### Tổng quan:

ExpressJS là một web framework tối giản và linh hoạt cho NodeJS, cung cấp các tính năng mạnh mẽ để phát triển các ứng dụng web và di động.



Hình 2. 20: Cấu trúc project sử dụng Express

#### Ưu và nhược điểm:

1. Ưu điểm

* Nhẹ, tối giản, dễ học và sử dụng.
* Hỗ trợ middleware.
* Dễ dàng quản lý các định tuyến (routes).
* Hỗ trợ đa dạng các template engine (handlebars, jade, ejs,..v.v)
* Hỗ trợ đa dạng các database (MySQL, MongoDB)

1. Nhược điểm

* Cấu trúc (structure) và quy ước (convention) chưa thật sự chặt chẽ.
* Chưa cung cấp được kiểm lỗi thời gian thực, dẫn đến dễ phát sinh lỗi.
* Middleware gây sự lạ lẫm cho người mới.

### ReactJS

#### Tổng quan:

ReactJS một thư viện JavaScript mã nguồn mở được thiết kế bởi Facebook để tạo ra những ứng dụng web một cách nhanh chóng và hiệu quả. Mục đích cốt lõi của ReactJS không chỉ khiến cho trang web phải thật mượt mà còn phải nhanh, khả năng mở rộng cao và đơn giản.

Sức mạnh của nó xuất phát từ việc tập trung vào các thành phần (component) riêng lẻ. Chính vì vậy, thay vì làm việc trên toàn bộ ứng dụng web, ReactJS cho phép một developer có thể phá vỡ giao diện người dùng phức tạp thành các component nhỏ hơn.

#### Ưu và nhược điểm:

1. Ưu điểm

* Dễ học, dễ sử dụng.
* Tạo web động (dynamic) dễ dàng hơn.
* Các tái sử dụng component.
* Cải thiện hiệu suất (performance) do virtual DOM.

1. Nhược điểm

* Documentation (tài liệu) còn ít.
* Sự khó khăn với người mới do ReactJS sử dụng JSX (điều này sự kết hợp giữa JSX và HTML).

### NodeJS

#### Tổng quan:

NodeJS là môi trường thời gian chạy (runtime environment) JavaScript đa nền tảng và mã nguồn mở. NodeJS cho phép các lập trình viên tạo cả ứng dụng front-end và back-end bằng JavaScript.

#### Ưu và nhược điểm:

1. Ưu điểm

* Mã nguồn mở.
* Hỗ trợ đa nền tảng.
* Dễ dàng mở rộng.
* Tốc độ thực thi nhanh.
* Truyền dữ liệu nhanh.
* Không có bộ nhớ đệm.
* Cộng đồng, tài liệu lớn.
* Tiết kiệm thời gian, chi phí, công sức.

1. Nhược điểm

* Chỉ có đơn luồng (single-threaded).
* Có khả năng gây bottleneck cho CPU.
* Các vấn đề phức tạp với bất đồng bộ (asynchronous).
* Phụ thuộc nhiều vào các bên thứ 3 (các thư viện bên ngoài).

# MÔ HÌNH ĐỀ XUẤT

## Các mô hình kết hợp

### BM-25 - BiLSTM

* Kết hợp giữa BM-25 và BiLSTM trong hệ thống ODQA.
* Sử dụng BM-25 để lấy context từ câu hỏi và tìm kiếm các từ khóa quan trọng.
* BiLSTM được sử dụng để phân tích ngữ cảnh chi tiết và tìm hiểu mối quan hệ giữa các từ khóa.
* Mô hình có khả năng nắm bắt sự tương tác phức tạp giữa các từ và biểu hiện ý nghĩa của câu hỏi.

### BM-25 - MRCQuestionAnswering

* Kết hợp giữa BM-25 và MRC Question Answering trong hệ thống ODQA.
* Sử dụng BM-25 để lấy context từ câu hỏi và định vị các từ khóa quan trọng.
* MRC Question Answering được sử dụng để đọc và hiểu nội dung của context và trả lời câu hỏi.
* Kết hợp khả năng định vị từ khóa của BM-25 và khả năng hiểu ngữ cảnh của MRC để cung cấp câu trả lời chính xác và toàn diện trong ODQA.

### BM-25 - PhoBERT

* Kết hợp giữa BM-25 và PhoBERT trong hệ thống ODQA.
* Sử dụng BM-25 để lấy context từ câu hỏi và định vị các từ khóa quan trọng.
* PhoBERT được sử dụng để nhúng và hiểu ngữ cảnh của context và câu hỏi, giúp mô hình có khả năng hiểu sâu và phức tạp.
* Tận dụng sức mạnh của định vị từ khóa từ BM-25 và khả năng hiểu ngữ cảnh của PhoBERT để cung cấp câu trả lời đầy đủ và hiệu quả trong ODQA.

## Mô hình không kết hợp

### GPT-2

* Sử dụng mô hình học sâu không kết hợp GPT-2 trong hệ thống ODQA.
* GPT-2 (Generative Pre-trained Transformer 2) được sử dụng để đọc và hiểu câu hỏi.
* Mô hình có khả năng tự sinh ra câu trả lời dựa trên kiến thức đã học từ tập dữ liệu lớn trước đó.
* Tận dụng khả năng tự sinh của GPT-2 để cung cấp câu trả lời mà không cần phải dựa vào các mô hình thống kê khác.

# THỰC NGHIỆM

## Dữ liệu thực nghiệm

* Số lượng dữ liệu: 2490 dòng. Dữ liệu được chia thành hai phần: 2440 dòng để huấn luyện và 50 dòng để kiểm tra.
* Phần train (2440 dòng): Bao gồm các cặp câu hỏi, bối cảnh và câu trả lời liên quan đến tuyển sinh. Đây là phần mà mô hình sẽ được huấn luyện dựa trên.
* Phần test (50 dòng): Bao gồm các cặp câu hỏi, bối cảnh và câu trả lời mà mô hình chưa được tiếp xúc trước đó. Phần này được sử dụng để đánh giá hiệu suất của mô hình sau khi đã được huấn luyện.
* Định dạng dữ liệu (.csv): Mỗi dòng trong tập dữ liệu chứa một câu hỏi, bối cảnh và câu trả lời, được phân tách bằng một ký tự đặc biệt hoặc một dấu phân cách cụ thể.
* Mục tiêu: Mục tiêu của bộ dữ liệu này là cung cấp một tập dữ liệu đa dạng về các câu hỏi và câu trả lời liên quan đến tuyển sinh, giúp huấn luyện và đánh giá mô hình AI về tuyển sinh.

## Cài đặt thực nghiệm

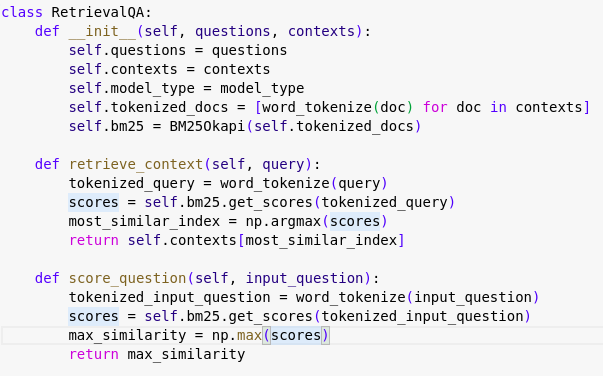
### Chuẩn bị Môi trường và cài đặt thư viện

* Hệ thống phần cứng: GPU T4 hoặc mạnh hơn
* Cài đặt môi trường: Sử dụng Jupyter Notebook hoặc Google Colab, cài đặt thư viện Pytorch và các thư viện quan trọng khác như pandas, numpy,...

### Xây dựng mô hình

#### Cài đặt mô hình BM-25

* Tải thư viện cần thiết: BM25Okapi
* Xây dựng lớp RetrievalQA:



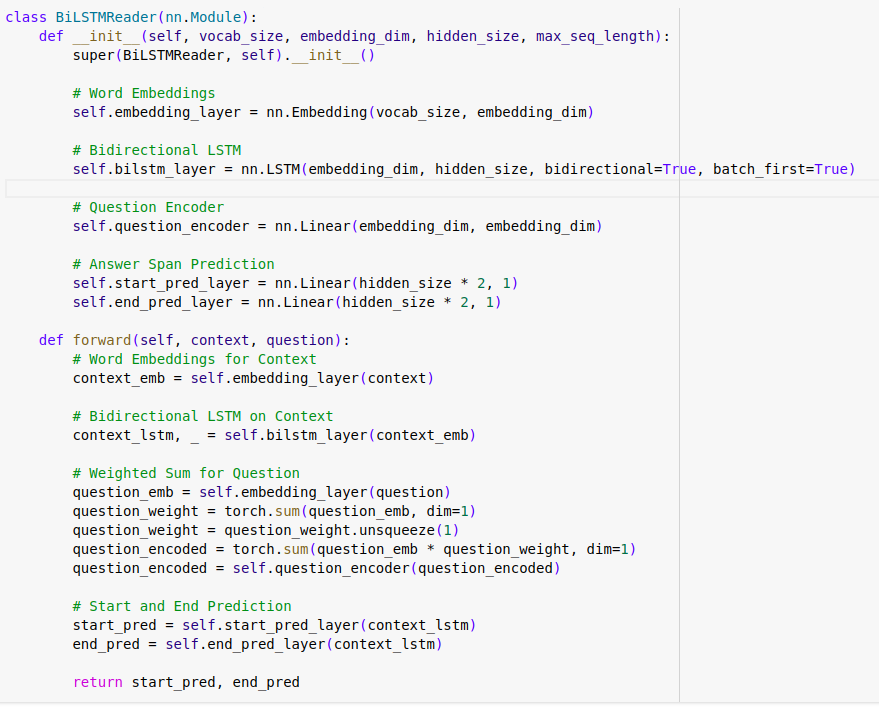
Hình 4.1 Code xây dựng mô hình BM-25

Phương thức retrieve\_context(self, query): Phương thức này nhận một câu hỏi (query) và trả về context tương ứng với câu hỏi đó. Để làm điều này, nó sử dụng mô hình BM25 để tính điểm cho mỗi context và chọn context có điểm cao nhất.

Phương thức score\_question(self, input\_question): Phương thức này nhận một câu hỏi (input\_question) và trả về điểm tương ứng cho câu hỏi đó. Nó tính điểm tương tự như retrieve\_context, nhưng thay vì trả về context, nó chỉ trả về điểm cao nhất.

#### Cài đặt mô hình BiLSTM

* Tải thư viện cần thiết: Pytorch
* Xây dựng lớp BiLSTM:



Hình 4.2 Code mô hình BiLSTM

Word Embeddings: Mô hình bắt đầu bằng việc ánh xạ từ vựng (vocab) vào không gian vector embedding với kích thước vocab\_size và embedding\_dim.

Bidirectional LSTM (BiLSTM): Sau đó, các từ được truyền qua một lớp BiLSTM để trích xuất thông tin ngữ cảnh từ context. Điều này giúp mô hình hiểu được mối quan hệ giữa các từ trong context theo cả hai hướng trước và sau.

Question Encoder: Câu hỏi được biểu diễn thông qua một lớp tuyến tính để tạo ra một vector biểu diễn cho câu hỏi.

Start và End Prediction: Cuối cùng, các câu trả lời được dự đoán thông qua hai lớp tuyến tính (start\_pred\_layer và end\_pred\_layer) dựa trên thông tin từ lớp BiLSTM.

#### Cài đặt mô hình MRC

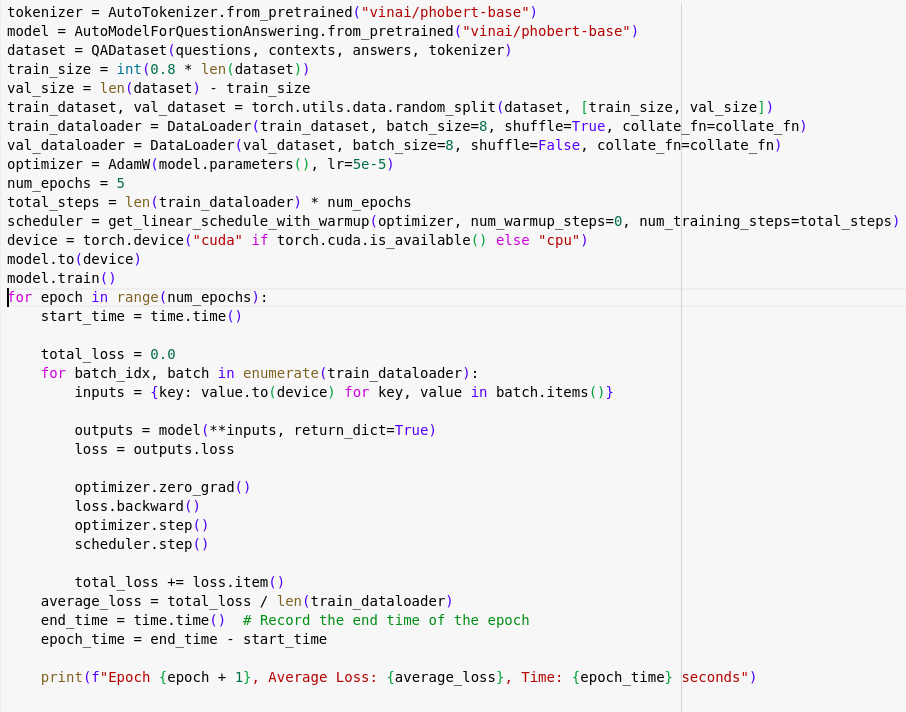
* Tải thư viện cần thiết: Transformers
* Tải project github để chạy mô hình:   
   https://github.com/nguyenvulebinh/extractive-qa-mrc
* Tiến hành sử dụng mô hình pre-training



Hình 4.3 Code sử dụng mô hình MRC

#### Cài đặt mô hình PhoBERT

* Tải thư viện cần thiết: Transformers
* Tiến hành tinh chỉnh mô hình trên tập dữ liệu đã chuẩn bị:

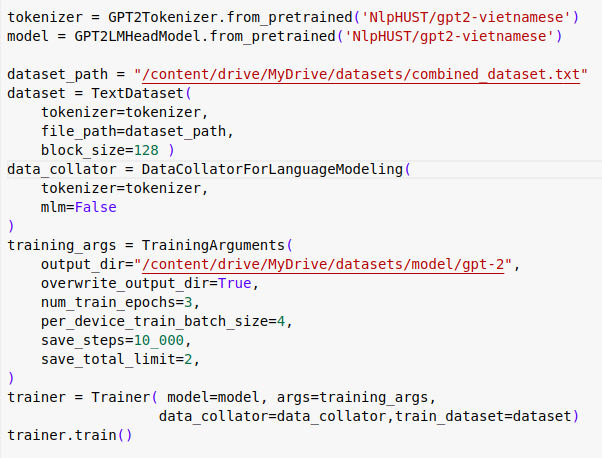


Hình 4.4 Code tinh chỉnh mô hình PhoBERT

Trong đoạn mã này, chúng ta sử dụng mô hình PhoBERT để huấn luyện một hệ thống trả lời câu hỏi (QA system). Đầu tiên, chúng ta tải tokenizer và mô hình từ thư viện transformers. Sau đó, chúng ta chuẩn bị dữ liệu đầu vào bằng cách sử dụng QADataset và chia thành tập huấn luyện và tập validation. Tiếp theo, chúng ta định nghĩa các tham số huấn luyện như batch size, optimizer và số epoch. Sau mỗi epoch, chúng ta huấn luyện mô hình trên tập huấn luyện, tính toán loss và cập nhật trọng số bằng cách sử dụng optimizer và scheduler.

#### Cài đặt mô hình GPT2

* Tải thư viện cần thiết: Transformers
* Tiến hành tinh chỉnh mô hình trên tập dữ liệu đã chuẩn bị:

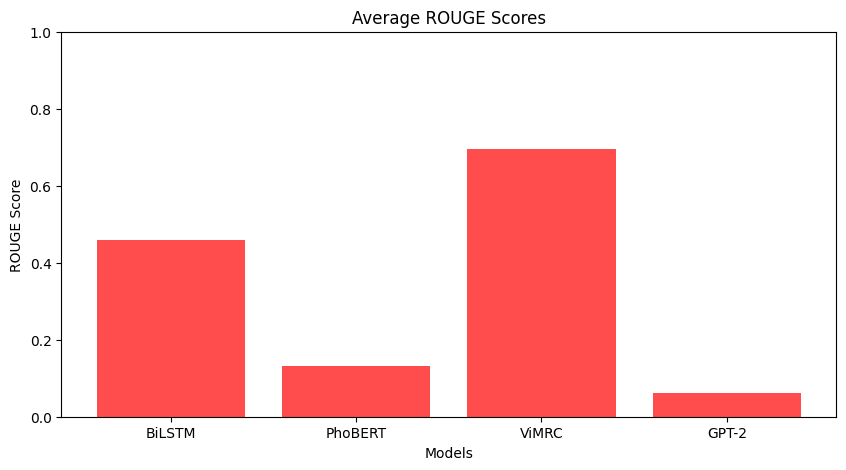


Hình 4.5 Code tinh chỉnh mô hình GPT-2

Trong đoạn mã trên, chúng ta sử dụng mô hình GPT-2 và tokenizer tương ứng để huấn luyện một mô hình ngôn ngữ cho tiếng Việt. Đầu tiên, chúng ta tải tokenizer và mô hình từ thư viện transformers. Sau đó, chúng ta chuẩn bị dữ liệu đầu vào bằng cách sử dụng TextDataset và chỉ định kích thước block. Tiếp theo, chúng ta thiết lập các tham số huấn luyện như đường dẫn lưu trữ, số epoch, kích thước batch và tần số lưu model. Cuối cùng, chúng ta sử dụng Trainer để huấn luyện mô hình, sử dụng data\_collator để chuẩn bị dữ liệu và các tham số huấn luyện đã thiết lập trước đó.

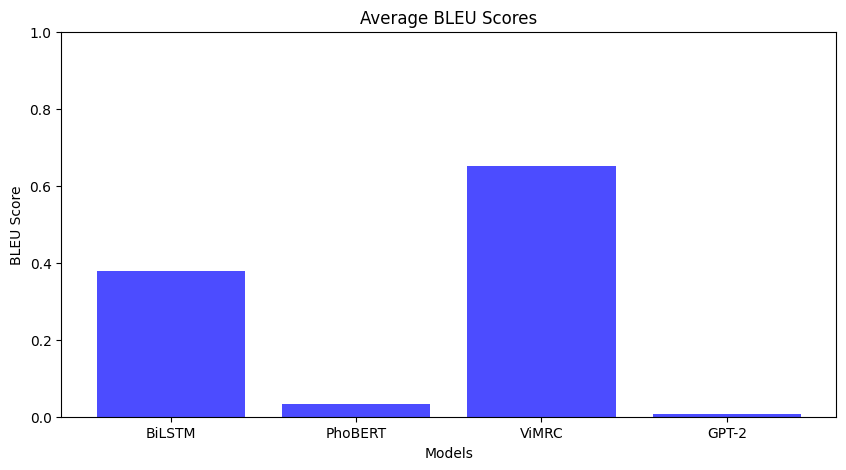
## Kết quả thực nghiệm

### Kết quả so sánh điểm ROUGE



Hình 4.6 Kết quả đánh giá mô hình bằng điểm ROUGE

### Kết quả so sánh điểm BLEU

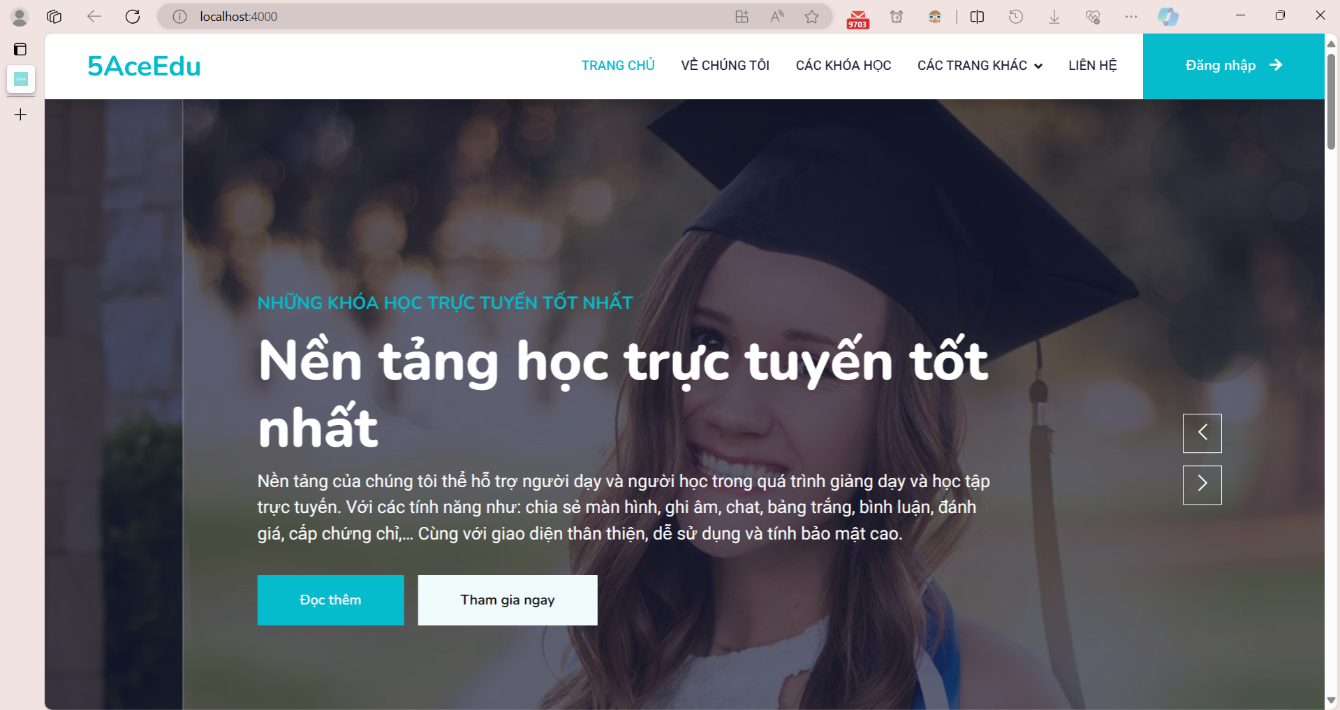


Hình 4.7 Kết quả đánh giá mô hình bằng điểm BLEU

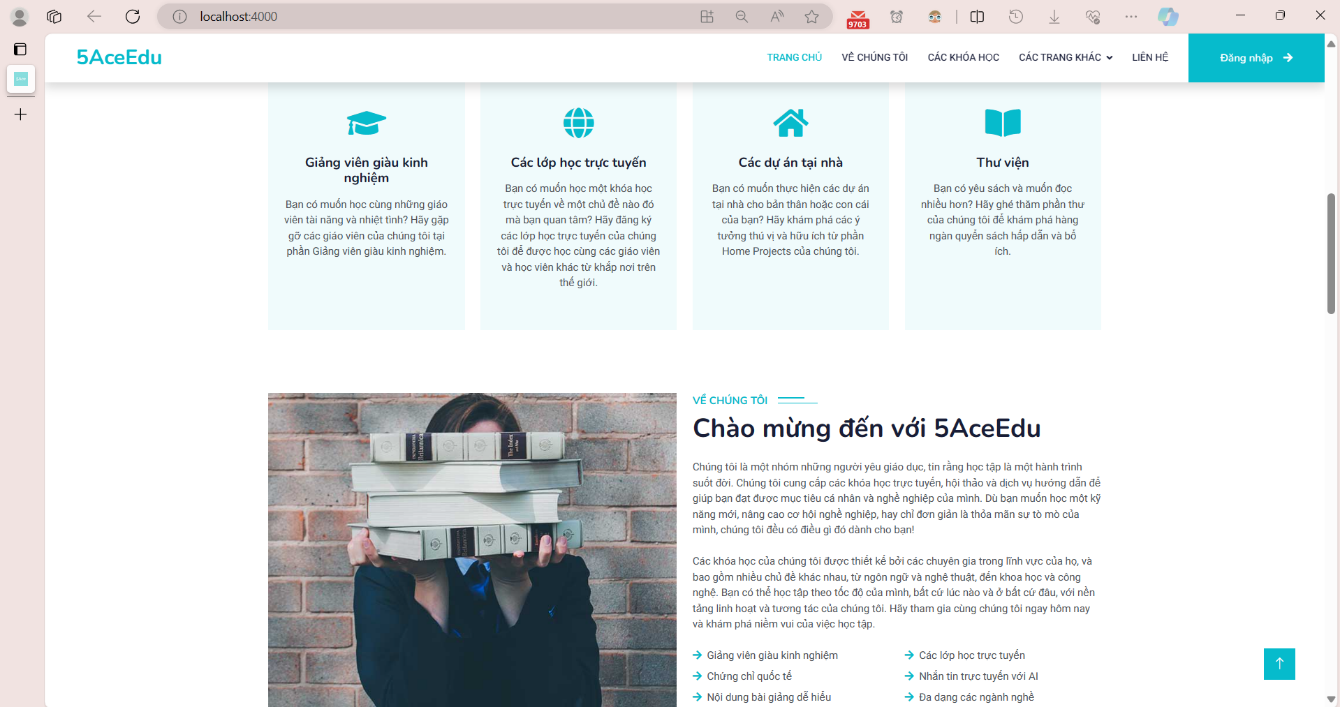
# KẾT QUẢ ĐẠT ĐƯỢC

## EndUser website module

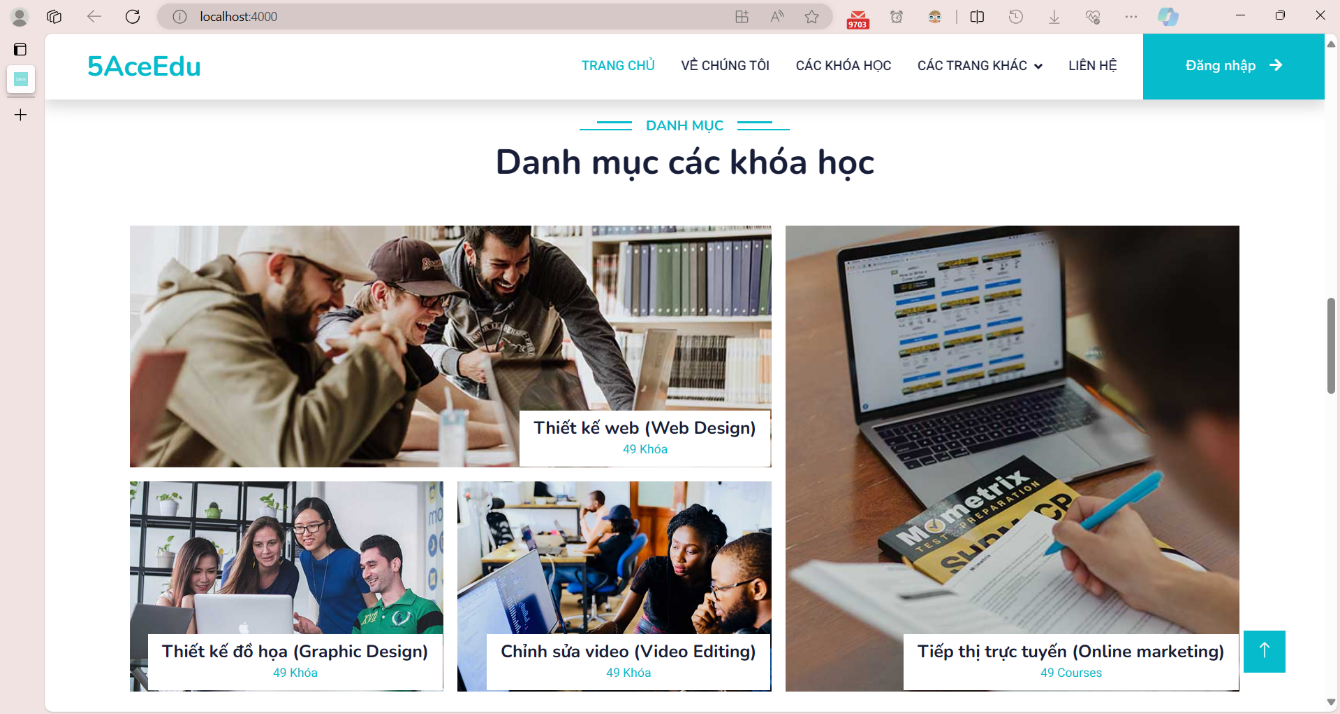
### Màn hình trang chủ



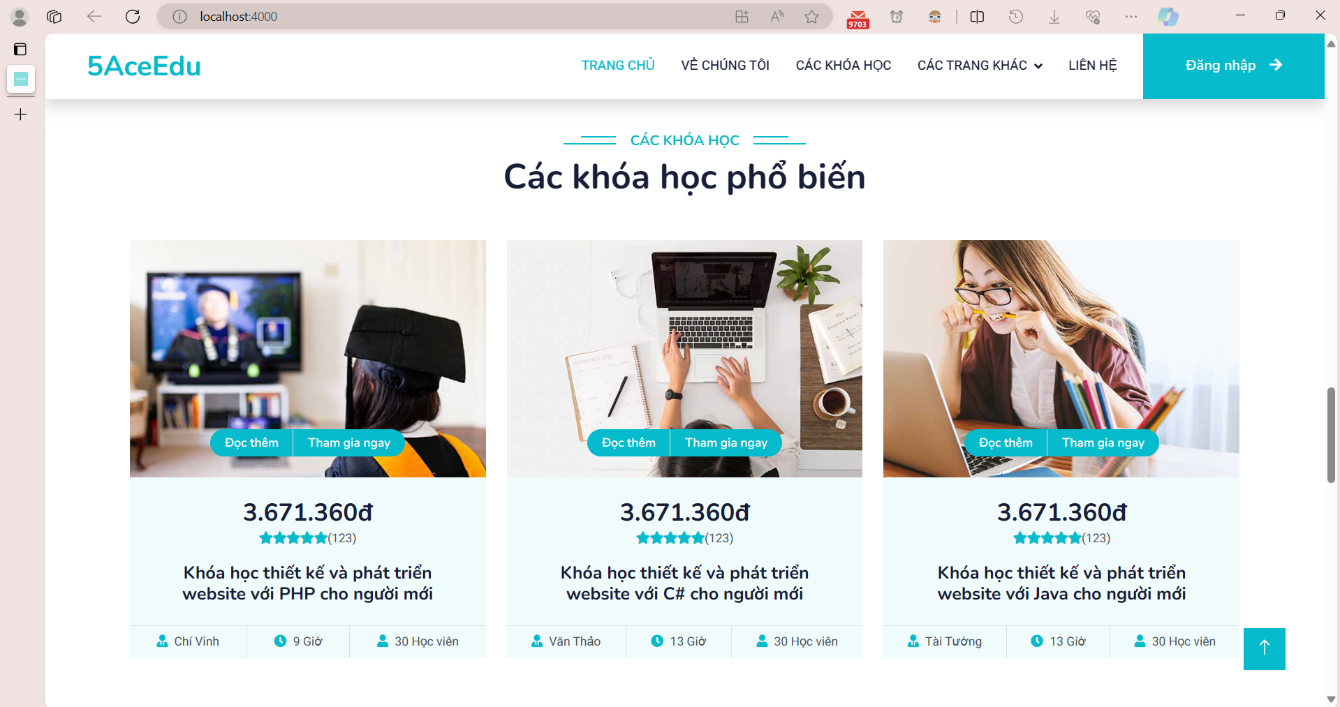
Hình 5.1: Section đầu trang của màn hình trang chủ.



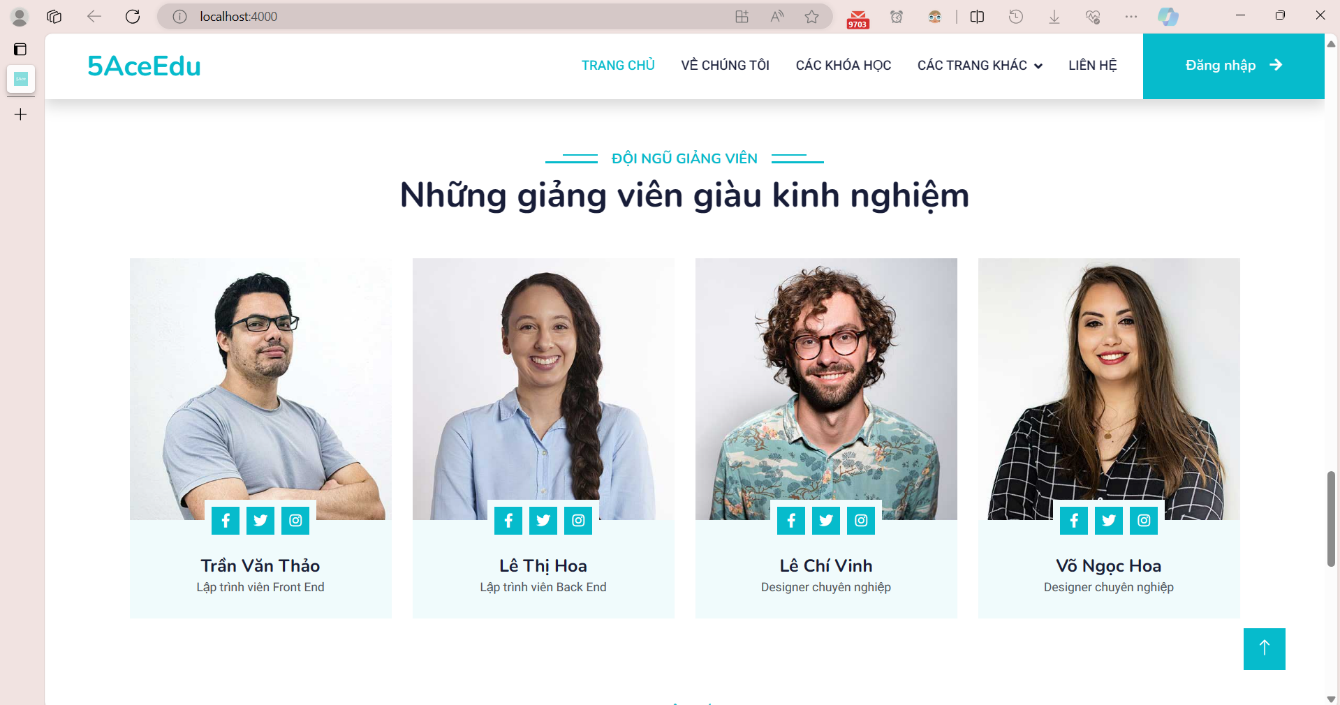
Hình 5.2: Section về chúng tôi và section các dịch vụ của màn hình trang chủ.



Hình 5.3: Section danh mục của màn hình trang chủ.



Hình 5.4: Section các khóa học của màn hình trang chủ.



Hình 5.5: Section đội ngũ giảng viên của màn hình trang chủ.

# KẾT LUẬN

## Kết luận

…

## Hướng phát triển

….

# TÀI LIỆU THAM KHẢO

Tiếng Việt

1. Phamdinhkhanh (May 2020). *Bài 36 - BERT model*. https://phamdinhkhanh.github.io/2020/05/23/BERTModel.html
2. Ban biên tập Topdev (2018). *MongoDB là gì? Định nghĩa đầy đủ và chi tiết nhất về MongoDB*. https://topdev.vn/blog/mongodb-la-gi/
3. Hoang Hung (March 2018). *Mongoose cho MongoDB, Nodejs*. https://viblo.asia/p/mongoose-cho-mongodb-nodejs-Qbq5QWvJZD8
4. Tuong Uyen (Dec 2023). *NodeJS là gì: Tổng quan kiến thức NodeJS và Top 5 NodeJS framework.* https://itviec.com/blog/nodejs-la-gi/

Tiếng Anh

1. Weng, Lilian. (Oct 2020). *How to build an open-domain question answering system?* Lil’Log. https://lilianweng.github.io/posts/2020-10-29-odqa/
2. Pearce, Kate & Alghowinem, Sharifa & Breazeal, Cynthia. (2022). *Build-a-Bot: Teaching Conversational AI Using a Transformer-Based Intent Recognition and Question Answering Architecture*.
3. Doug Turnbull (Oct 2015). *BM25 The Next Generation of Lucene Relevance.* https://opensourceconnections.com/blog/2015/10/16/bm25-the-next-generation-of-lucene-relevation/
4. Christopher Olah (Au 2015). *Understanding LSTM Networks*. https://colah.github.io/posts/2015-08-Understanding-LSTMs/
5. aditya\_taparia (Jun 2023). *Bidirectional LSTM in NLP.* https://www.geeksforgeeks.org/bidirectional-lstm-in-nlp/
6. Rani Horev (Nov 2018). *BERT Explained: State of the art language model for NLP.* Towardsdatascience. https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270
7. Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova (May 2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* (arXiv:1810.04805v2). arXiv. https://doi.org/10.48550/arXiv.1810.04805
8. Radford, Alec, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever (2018). *Improving language understanding by generative pre-training*. https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language\_understanding\_paper.pdf
9. DataFlair contributor (2023), *Advantages and Disadvantages of ExpressJS. data-flair. https://data-flair.training/blogs/expressjs-advantages-and-disadvantages/*
10. EPAM Anywhere Editorial (Feb 2024). *top 5 NodeJS pros and cons: what they mean for your project*. anywhere. https://anywhere.epam.com/en/blog/node-js-pros-and-cons