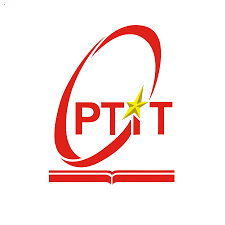
**BỘ THÔNG TIN TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

-----🙞🙜🕮🙞🙜-----

**BÁO CÁO BÀI TẬP LỚN**

**Đề tài: Nhận dạng thái độ của các đoạn văn bản sử dụng mô hình**

**PhoBERT và BiLSTM**

**Môn học: Phát triển hệ thống thông minh**

**Nhóm học phần: 04 - Nhóm BTL: 02**

|  |  |
| --- | --- |
| **Trần Xuân Đạt** | **B21DCCN223** |
| **Phạm Tuấn Giang** | **B21DCCN764** |
| **Lê Đức Nam** | **B21DCCN547** |
| **Trần Việt Anh** | **B21DCCN047** |
| **Lương Mạnh Hoà** | **B21DCCN378** |

**Hà Nội – 11/2024**

# Cám ơn

Trong quá trình nghiên cứu và hoàn thành báo cáo bài tập lớn của môn Phát triển Hệ thống thông minh, bên cạnh sự nỗ lực của các thành viên trong nhóm, chúng tôi cũng đã nhận được sự giảng dạy và hướng dẫn nhiệt tình của các thầy cô giáo khoa Công nghệ thông tin. Chúng tôi xin gửi lời cảm ơn chân thành tới thầy Trần Trung Tuấn đã tận tình, chu đáo hướng dẫn nhóm chúng tôi trong suốt quá trình học tập, nghiên cứu để chúng tôi có thể hoàn thành đề tài “ Nhận dạng thái độ của các đoạn văn bản sử dụng mô hình PhoBERT và BiLSTM”.

Nhóm chúng tôi cũng xin được cảm ơn người thân và gia đình đã quan tâm, động viên và luôn tjao cho những thành viên trong nhóm những điều kiện tốt nhất trong suốt quá trình học tập và thực hiện đề tài này.

Ngoài ra chúng tôi xin gửi lời cảm ơn tới tất cả bạn bè, những người bạn trong lớp đã luôn gắn bó, cùng họp tập và giúp đỡ chúng tôi trong suốt quá trình học tập và thực hiện bài tập lớn.

Trong suốt quá trình thực hiện bài tập lớn, cũng như trong quá trình làm bài cáo cáo, do trình độ lý luận cũn như kinh nghiệm thực tiễn vẫn còn hạn chế nên chúng tôi khó tránh khỏi những sai sót, rất mong mọi người bỏ qua và chúng tôi rất mong muốn nhận được những đóng góp ý kiến của các thầy cô và những nhà nghiên cứu khác để nội dung nghiên cứu được hoàn thiện hơn.

# Mục lục

[Cám ơn ii](#_Toc182950480)

[Mục lục iii](#_Toc182950481)

[Danh sách các từ viết tắt vi](#_Toc182950482)

[Danh sách các hình vẽ, bảng biểu viii](#_Toc182950483)

[Mở đầu 1](#_Toc182950484)

[1. Lý do chọn đề tài 1](#_Toc182950485)

[2. Cấu trúc báo cáo 2](#_Toc182950486)

[Chương 1. Cơ sở lý thuyết 3](#_Toc182950487)

[1.1. Học máy 3](#_Toc182950488)

[1.1.1 Khái niệm và phân loại 3](#_Toc182950489)

[1.1.2 Các loại hình học máy 3](#_Toc182950490)

[1.1.3. Ứng dụng của học máy 3](#_Toc182950491)

[1.2 Học sâu 4](#_Toc182950492)

[1.2.1 Khái niệm học sâu 4](#_Toc182950493)

[1.2.2 Các kiến trúc mạng nơ-ron phổ biến trong học sâu 4](#_Toc182950494)

[1.2.3 Ứng dụng của học sâu 5](#_Toc182950495)

[1.2.4. Thách thức của học sâu 6](#_Toc182950496)

[1.3 Bài toán phân loại 6](#_Toc182950497)

[1.3.1. Giới thiệu 6](#_Toc182950498)

[1.3.2. Định nghĩa bài toán 6](#_Toc182950499)

[1.3.3. Các loại bài toán phân loại 7](#_Toc182950500)

[1.3.4. Các loại thuật toán phân loại phổ biến 7](#_Toc182950501)

[1.4 Mô hình phoBERT 8](#_Toc182950502)

[1.4.1. Giới thiệu về BERT 8](#_Toc182950503)

[1.4.2. PhoBERT là gì? 9](#_Toc182950504)

[1.4.3. Cách thức PhoBERT hoạt động 9](#_Toc182950505)

[1.5 Mô hình BiLSTM 10](#_Toc182950506)

[1.5.1. Định nghĩa 10](#_Toc182950507)

[1.5.2. Kiến trúc lớp BiLSTM 11](#_Toc182950508)

[1.6. Kết luận 13](#_Toc182950509)

[Chương 2. Thử nghiệm 14](#_Toc182950510)

[2.1 Bộ dữ liệu 14](#_Toc182950511)

[2.1.1 Tổng quan dữ liệu 14](#_Toc182950512)

[2.1.2 Các tập dữ liệu 14](#_Toc182950513)

[2.3 Các phân tích thống kê về bộ dữ liệu và đồ họa 15](#_Toc182950514)

[2.3 Mô hình 20](#_Toc182950515)

[2.3.1 Tiền xử lý: 20](#_Toc182950516)

[2.3.2 Quy trình xây dựng mô hình: 24](#_Toc182950517)

[2.3.3 Xây dựng mô hình phoBERT. 24](#_Toc182950518)

[Một số thuộc tính sử dụng trong quá trình training: 25](#_Toc182950519)

[Thiết lập tham số huấn luyện 26](#_Toc182950520)

[2.3.4 Xây dựng mô hình BiLSTM 28](#_Toc182950521)

[2.3.5 Kết Quả Cuối Cùng 29](#_Toc182950522)

[a. Mô hình phoBERT 29](#_Toc182950523)

[b. Mô hình BiLSTM 30](#_Toc182950524)

[c. Kiểm tra hai mô hình với các câu mới 31](#_Toc182950525)

[2.4 Kết luận 33](#_Toc182950526)

[a. So sánh kết quả 33](#_Toc182950527)

[b. So sánh kích thước model 33](#_Toc182950528)

[c. Đánh giá 33](#_Toc182950529)

[Chương 3: Kết luận 35](#_Toc182950530)

[1. Đã làm được 35](#_Toc182950531)

[2. Phương hướng tương lai 36](#_Toc182950532)

[Tài liệu tham khảo 38](#_Toc182950533)

# Danh sách các từ viết tắt

|  |  |  |
| --- | --- | --- |
| **Viết tắt** | **Tiếng Anh** | **Giải nghĩa** |
| BERT | Bidirectional Encoder Representations from Transformers | Mô hình ngôn ngữ dùng để hiểu ngữ cảnh của từ vựng trong văn bản bằng cách sử dụng kiến trúc Transformers hai chiều |
| BiLSTM | Bidirectional Long Short-Term Memory | Mạng LSTM hai chiều, là một dạng RNN cải tiến có khả năng ghi nhớ thông tin lâu hơn và phân tích dữ liệu theo cả hai hướng |
| BPE | Byte-Pair Encoding | Phương pháp mã hóa cặp byte, một kỹ thuật để phân đoạn từ ngữ thành các đơn vị nhỏ hơn |
| CNN | Convolutional Neural Network | Mạng nơ-ron tích chập, chuyên được sử dụng trong xử lý ảnh và video nhờ vào khả năng phát hiện các đặc trưng của dữ liệu không gian, chẳng hạn như cạnh và hình dạng |
| GPU | Graphic Processing Unit | Đơn vị xử lý đồ họa, thiết bị phần cứng chuyên dụng cho việc xử lý các tính toán phức tạp nhanh chóng. |
| GRU | Gated Recurrent Unit | Đơn vị hồi tiếp có cổng, một biến thể của LSTM giúp đơn giản hóa cấu trúc và tăng tốc độ tính toán |
| KNN | K-Nearest Neighbors | Phương pháp "hàng xóm gần nhất", là một thuật toán phân loại không tham số dựa trên khoảng cách |
| LSTM | Long Short-Term Memory | Mạng nơ-ron bộ nhớ ngắn hạn dài, một loại RNN được thiết kế để giải quyết vấn đề biến mất của gradient trong việc học các chuỗi dài hạn. |
| ML | Machine Learning | Học máy tập trung vào việc cho phép máy tính học từ dữ liệu để thực hiện các nhiệm vụ. |
| NLP | Natural Language Processing | Xử lý ngôn ngữ tự nhiên, lĩnh vực nghiên cứu cho phép máy tính hiểu, phân tích và tạo ra ngôn ngữ con người |
| RNN | Recurrent Neural Network | Mạng nơ-ron hồi quy, có các vòng lặp cho phép thông tin lưu thông và ghi nhớ ngữ cảnh của dữ liệu chuỗi. |
| SVM | Support Vector Machine | Máy vector hỗ trợ, là thuật toán phân loại có mục tiêu tạo ra một siêu phẳng phân tách dữ liệu một cách tối ưu nhất. |
| TPU | Tensor Processing Unit | Đơn vị xử lý tensor, phần cứng chuyên dụng được Google phát triển cho việc xử lý các mô hình học sâu lớn trên nền tảng TensorFlow. |

# Danh sách các hình vẽ, bảng biểu

[Hình 1.1 Cấu trúc mạng nơ-ron nhân tạo](#_1y810tw) 4

[Bảng 2.1. Dữ liệu thí dụ](#_37m2jsg) 14

[Hình 2.1. Biểu đồ tỉ lệ cảm xúc các câu trong bộ dữ liệu train](#_46r0co2) 15

[Hình 2.2 Biểu đồ tỉ lệ cảm xúc các câu trong bộ dữ liệu test 16](#_Hình_2.2._Biểu)

[Hình 2.3 Biểu đồ tỉ lệ cảm xúc các câu trong bộ dữ liệu valid 16](#_Hình_2.3._Biểu)

[Hình 2.4 Biểu đồ thể hiện phân bố độ dài câu 17](#_Hình_2.4_Biểu)

[Hình 2.5 Biểu đồ thể hiện tần suất từ xuất hiện trong nhãn tiêu cực 18](#_Hình_2.5_Biểu)

[Hình 2.6 Biểu đồ thể hiện tần suất từ xuất hiện trong nhãn bình thường 18](#_Hình_2.6_Biểu)

[Hình 2.7 Biểu đồ thể hiện tần suất từ xuất hiện trong nhãn tích cực 19](#_Hình_2.7_Biểu)

[Hình 2.8 Nạp mô hình phoBERT 24](#_Hình_2.8_Nạp)

[Hình 2.9. Mô tả cho mô hình BiLSTM 27](#_Hình_2.9_Mô)

[Hình 2.10 Kết quả đánh giá độ chính xác mô hình phoBERT 28](#_Hình_2.10_Kết_1)

[Hình 2.11 Kết quả dữ đoán đúng của từng nhãn 29](#_Hình_2.10_Kết)

[Hình 2.12 Kết quả dữ đoán đối với mô hình BiLSTM 29](#_Hình_2.11_Kết)

[Hình 2.13 Kết quả dự đoán đối với từng nhãn của mô hình BiLSTM 30](#_Hình_2.13_Kết)

[Hình 2.14 Kết quả dự đoán tập câu mẫu của PhoBERT 30](#_Hình_2.14_Kết)

[Hình 2.15 Kết quả dự đoán tập test của PhoBERT 31](#_Hình_2.15_Kết)

[Hình 2.16 Kết quả dự đoán tập mẫu của BiLSTM 3](#_Hình_2.16_Kết)1

[Hình 2.17 Kết quả dự đoán tập test của BiLSTM 3](#_Hình_2.16_Kết)2

[Bảng 2.2. Thiết lập tham số huấn luyện](#_Bảng_2.2:_Thiết) 26

[Bảng 2.3 So sánh kết quả hai mô hình](#_Bảng_2.2_So) 32

[Bảng 2.4 So sánh kích thước hai mô hình](#_Bảng_2.3._So) 32

# Mở đầu

## 1. Lý do chọn đề tài

Với sự phát triển không ngừng các lĩnh vực công nghệ, việc nhận diện cảm xúc trong văn bản tiếng Việt được ứng dụng trong nhiều lĩnh vực như: quản trị doanh nghiệp, quản trị thương hiệu sản phẩm, quản trị quan hệ khách hàng, khảo sát ý kiến khách hàng hay dễ hiểu hơn là phân tích đánh giá, ý kiến phản hồi của khách hàng về một sản phẩm, dịch vụ, …. Việc dự đoán là vô cùng quan trọng vì ý kiến, đánh giá của khách hàng ngày càng trở nên có giá trị thiết thực hơn.

Mục tiêu của bài toán phân loại cảm xúc văn bản là xây dựng một hệ thống tự động có khả năng nhận diện và phân loại cảm xúc trong các đoạn văn bản. Cụ thể, hệ thống này sẽ:

* Phân loại cảm xúc: Xác định cảm xúc của văn bản theo các nhãn cảm xúc đã định trước, chẳng hạn như tích cực, tiêu cực, trung lập hoặc các cảm xúc chi tiết hơn như vui, buồn, giận dữ, sợ hãi, ngạc nhiên, v.v.
* Hiểu ngữ cảnh: Đảm bảo rằng việc nhận diện cảm xúc không chỉ dựa trên từ khoá mà còn hiểu được ngữ cảnh của câu để đưa ra đánh giá chính xác hơn.
* Tự động hóa phân tích: Giảm thiểu sự cần thiết của con người trong việc phân tích và đánh giá cảm xúc của lượng lớn văn bản, từ đó tăng hiệu quả và tiết kiệm thời gian.
* Ứng dụng rộng rãi: Áp dụng trong nhiều lĩnh vực như phân tích phản hồi khách hàng, đánh giá sản phẩm, giám sát mạng xã hội, hỗ trợ dịch vụ khách hàng, và nhiều ứng dụng khác liên quan đến phân tích cảm xúc trong ngôn ngữ tự nhiên.
* Nâng cao trải nghiệm người dùng: Giúp các doanh nghiệp và tổ chức hiểu rõ hơn về cảm xúc và quan điểm của khách hàng, từ đó cải thiện dịch vụ và sản phẩm, nâng cao trải nghiệm của người dùng.

## 2. Cấu trúc báo cáo

Báo cáo này gồm 4 phần:

* **Chương 1. Cơ sở lý thuyết:** Giới thiệu khái niệm và tìm hiểu về học máy/học sâu là gì, giới thiệu về bài toán phân loại, các mô hình nhóm sử dụng trong bài toán thực tế.
* **Chương 2. Thử nghiệm mô hình:** Giới thiệu và khai phá về bộ dữ liệu nhóm sử dụng, quá trình tiền xử lý mô hình, quá trình xây dựng, quá trình huấn luyện và kết quả.
* **Chương 3. Kết quả và kết luận:** Những điều nhóm đã làm được trong quá trình giải quyết bài toán và phương hướng phát triển bài toán trong tương lai.
* **Tài liệu tham khảo**

# Chương 1. Cơ sở lý thuyết

## 1.1. Học máy

### 1.1.1 Khái niệm và phân loại

Học máy (Machine Learning) là một nhánh của khoa học máy tính, nơi các hệ thống và thuật toán có thể học hỏi từ dữ liệu mà không cần lập trình cụ thể từng quy tắc. Mô hình học máy sẽ tự động tìm ra các mẫu và mối quan hệ ẩn trong dữ liệu để dự đoán hoặc ra quyết định dựa trên dữ liệu đầu vào. Quá trình này giúp mô hình cải thiện dần hiệu suất qua thời gian, dựa trên kinh nghiệm từ dữ liệu đã qua xử lý.

### 1.1.2 Các loại hình học máy

Học máy được chia thành ba loại chính:

* **Học có giám sát (Supervised Learning)**: Học có giám sát sử dụng dữ liệu đã gán nhãn, nơi các đầu vào và đầu ra mong muốn được cung cấp sẵn. Mô hình học từ các cặp đầu vào-đầu ra để dự đoán kết quả cho các dữ liệu mới. Các bài toán điển hình của học có giám sát là phân loại (classification) và hồi quy (regression).

Ví dụ: phân loại email là thư rác hay thư quan trọng dựa trên các mẫu đã gán nhãn.

* **Học không giám sát (Unsupervised Learning)**: Học không giám sát làm việc với dữ liệu không có nhãn, mô hình tự động tìm kiếm cấu trúc ẩn và các nhóm trong dữ liệu. Phương pháp này thường dùng trong các bài toán như phân cụm (clustering) và giảm chiều (dimensionality reduction).

Ví dụ: phân cụm khách hàng dựa trên thói quen mua sắm.

* **Học tăng cường (Reinforcement Learning)**: Trong học tăng cường, mô hình học qua việc tương tác với môi trường, nhận các phản hồi dưới dạng phần thưởng hoặc hình phạt từ các hành động thực hiện. Phương pháp này thường áp dụng cho các tác vụ như trò chơi, điều khiển robot, và tối ưu hóa chuỗi hành động trong các hệ thống phức tạp.

### 1.1.3. Ứng dụng của học máy

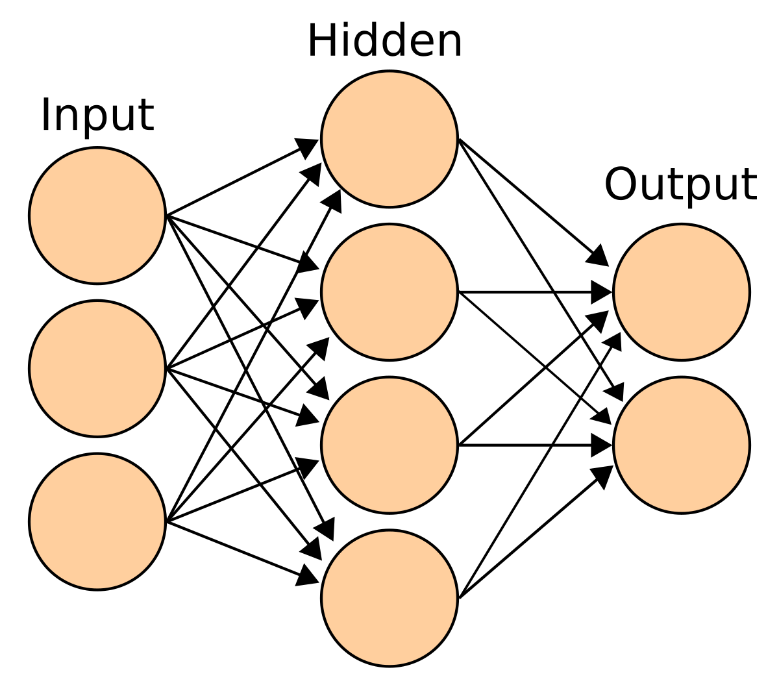
Học máy có thể được ứng dụng rộng rãi trong nhiều lĩnh vực, bao gồm:

* Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP): Phân tích cảm xúc, dịch tự động, và nhận diện ngữ nghĩa trong văn bản là những ứng dụng phổ biến của học máy.
* Thị giác máy tính (Computer Vision): Các mô hình học máy có khả năng nhận diện, phân loại, và theo dõi đối tượng trong hình ảnh và video. Ví dụ: nhận diện khuôn mặt, phân loại ảnh, và phát hiện sự kiện trong video.
* Dự đoán và chẩn đoán y khoa: Sử dụng dữ liệu y khoa để dự đoán các bệnh lý và chẩn đoán bệnh sớm từ hình ảnh y khoa hoặc hồ sơ sức khỏe điện tử.

## 1.2 Học sâu

### 1.2.1 Khái niệm học sâu

Học sâu (Deep Learning) là một nhánh của học máy, nơi các mô hình mạng nơ-ron nhân tạo (Artificial Neural Networks) với nhiều tầng ẩn được xây dựng và tối ưu hóa. Mạng nơ-ron sâu có khả năng học các đặc trưng phức tạp trong dữ liệu, đặc biệt hiệu quả khi làm việc với dữ liệu phi cấu trúc như văn bản, hình ảnh, và âm thanh. Học sâu đã trở thành phương pháp chủ đạo trong các nghiên cứu và ứng dụng về trí tuệ nhân tạo nhờ vào khả năng mở rộng và cải thiện độ chính xác trong nhiều bài toán phức tạp.



#### Hình 1.1 Cấu trúc mạng nơ-ron nhân tạo

### 1.2.2 Các kiến trúc mạng nơ-ron phổ biến trong học sâu

Trong học sâu, các kiến trúc mạng nơ-ron được xây dựng và tối ưu để phục vụ cho các bài toán cụ thể:

* **Mạng nơ-ron tích chập (Convolutional Neural Network - CNN)**: CNN được thiết kế để xử lý hình ảnh và dữ liệu không gian. Các tầng tích chập (convolutional layers) giúp mô hình tự động học các đặc trưng không gian từ thấp đến cao, như cạnh, góc, và các mẫu phức tạp hơn. CNN được ứng dụng rộng rãi trong phân loại hình ảnh, phát hiện đối tượng, và xử lý hình ảnh y tế.
* **Mạng nơ-ron hồi quy (Recurrent Neural Network - RNN)**: RNN được thiết kế để làm việc với dữ liệu chuỗi, ví dụ như văn bản, âm thanh, và dữ liệu thời gian. Nhờ vào khả năng lưu trữ thông tin từ các bước trước đó trong chuỗi, RNN có thể nắm bắt các phụ thuộc dài hạn trong dữ liệu chuỗi. Tuy nhiên, RNN truyền thống thường gặp vấn đề về độ dài chuỗi, mà các mô hình cải tiến như *LSTM (Long Short-Term Memory)* và *GRU (Gated Recurrent Unit)* có thể khắc phục, cho phép mô hình xử lý các chuỗi dài một cách hiệu quả hơn.
* **Transformer**: Transformer là kiến trúc mạng dựa trên cơ chế *self-attention* (tự chú ý), giúp xử lý hiệu quả các chuỗi dữ liệu dài và phức tạp mà không cần duyệt tuần tự. Kiến trúc này đã cách mạng hóa nhiều ứng dụng trong xử lý ngôn ngữ tự nhiên (NLP), chẳng hạn như dịch máy, phân tích ngữ nghĩa, và sinh văn bản. Các mô hình tiên tiến như *BERT* và *PhoBERT* (dành riêng cho tiếng Việt) đều dựa trên kiến trúc Transformer và có khả năng học được ngữ cảnh tốt hơn từ dữ liệu văn bản.

### 1.2.3 Ứng dụng của học sâu

Học sâu mở ra nhiều ứng dụng tiên tiến, cụ thể như:

* **Thị giác máy tính**: Học sâu có thể tự động học các đặc trưng phức tạp từ hình ảnh và video để phát hiện đối tượng, nhận diện khuôn mặt, và tạo ảnh tự động. CNN và Transformer đóng vai trò quan trọng trong các ứng dụng này.
* **Xử lý ngôn ngữ tự nhiên (NLP)**: Học sâu giúp cải thiện đáng kể các tác vụ như phân loại cảm xúc, phân tích ngữ cảnh, dịch máy, và tóm tắt văn bản. Các mô hình Transformer như BERT và PhoBERT đã tạo ra những bước tiến lớn trong việc xử lý và phân tích ngữ nghĩa trong ngôn ngữ.
* **Xe tự hành và robot**: Học sâu giúp các hệ thống xe tự hành phân tích hình ảnh từ camera và tín hiệu cảm biến, đưa ra quyết định trong thời gian thực để đảm bảo an toàn. Robot sử dụng học sâu để nhận diện đối tượng, phát hiện chướng ngại vật, và cải thiện khả năng tương tác với môi trường.

### 1.2.4. Thách thức của học sâu

Mặc dù mang lại nhiều tiềm năng, học sâu cũng đối mặt với các thách thức đáng kể:

* **Yêu cầu tài nguyên lớn**: Mô hình học sâu cần dữ liệu lớn và thường yêu cầu phần cứng mạnh như GPU hoặc TPU để huấn luyện hiệu quả.
* **Khó khăn trong việc điều chỉnh mô hình**: Quá trình tối ưu hóa mô hình học sâu đòi hỏi kiến thức chuyên sâu và kinh nghiệm về siêu tham số, khiến quá trình phát triển trở nên phức tạp.
* **Hạn chế về khả năng giải thích**: Các mô hình học sâu phức tạp thường là các "hộp đen," khó hiểu và giải thích, điều này có thể là trở ngại trong các ứng dụng yêu cầu tính minh bạch và giải trình như y tế hoặc tài chính.

## 1.3 Bài toán phân loại

### 1.3.1. Giới thiệu

Phân loại (classification) là một trong những bài toán cơ bản và phổ biến trong *machine learning* (ML), nơi mà mục tiêu là dự đoán nhãn (class) của một đối tượng dữ liệu dựa trên các đặc trưng đầu vào. Các ứng dụng của bài toán phân loại trải dài trên nhiều lĩnh vực như nhận dạng hình ảnh, xử lý ngôn ngữ tự nhiên, phân tích văn bản, và dự báo thị trường tài chính.

### 1.3.2. Định nghĩa bài toán

Bài toán phân loại có thể được định nghĩa là việc tìm kiếm một ánh xạ từ không gian đầu vào đến không gian các nhãn. Giả sử rằng ta có một tập dữ liệu huấn luyện gồm các cặp (x, y), trong đó:

* **x** là vector đặc trưng đại diện cho các thuộc tính của đối tượng (ví dụ: hình ảnh, văn bản, hoặc bảng dữ liệu).
* **y** là nhãn (label) tương ứng với đối tượng, được chọn từ một tập hữu hạn các lớp C={c1,c2,...,cn}
* Nhiệm vụ của thuật toán phân loại là học một hàm ánh xạ f:X→Y sao cho với bất kỳ đối tượng mới nào xnew​, thuật toán có thể dự đoán chính xác nhãn ynew

### 1.3.3. Các loại bài toán phân loại

Bài toán phân loại trong machine learning có thể chia thành các loại khác nhau dựa trên số lượng lớp và bản chất của nhãn:

* Phân loại nhị phân (Binary classification): Đây là bài toán phân loại với hai nhãn, ví dụ như phân loại email là spam hoặc không spam.
* Phân loại đa lớp (Multiclass classification): Bài toán phân loại có nhiều hơn hai lớp, ví dụ như phân loại hình ảnh thành các loại khác nhau như chó, mèo, chim.
* Phân loại đa nhãn (Multilabel classification): Mỗi đối tượng có thể thuộc về nhiều lớp khác nhau, ví dụ một bài báo khoa học có thể thuộc về nhiều danh mục như "Trí tuệ nhân tạo", "Xử lý ngôn ngữ tự nhiên", và "Máy học".

### 1.3.4. Các loại thuật toán phân loại phổ biến

1. K-Nearest Neighbors (KNN): KNN dựa trên nguyên tắc: một điểm dữ liệu sẽ được phân loại dựa trên nhãn của k điểm dữ liệu láng giềng gần nhất.

2. Support Vector Machine (SVM): SVM tìm cách xác định một siêu phẳng phân chia các lớp trong không gian đặc trưng.

3. Decision Trees (Cây quyết định): Cây quyết định phân chia dữ liệu thành các tập con dựa trên các điều kiện quyết định (decision rules). Đây là thuật toán dễ hiểu và dễ trực quan hóa.

4. Logistic Regression: Mặc dù có tên là "hồi quy", Logistic Regression thực chất là một thuật toán phân loại nhị phân. Nó sử dụng hàm sigmoid để dự đoán xác suất một đối tượng thuộc về một lớp nhất định.

5. Neural Networks (Mạng nơ-ron): Mạng nơ-ron sâu, đã mang lại những tiến bộ vượt bậc trong nhiều bài toán phân loại như nhận dạng hình ảnh, nhận dạng giọng nói, và phân loại văn bản. Mạng nơ-ron mô phỏng quá trình hoạt động của não bộ, với nhiều lớp nơ-ron kết nối.

## 1.4 Mô hình phoBERT

### 1.4.1. Giới thiệu về BERT

**1.4.1.1: BERT là gì?**

BERT (Bidirectional Encoder Representations from Transformers) là một mô hình học sẵn hay còn gọi là pre-train model, học ra các vector đại diện theo ngữ cảnh 2 chiều của từ, được sử dụng để transfer sang các bài toán khác trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP).

* NLP sử dụng một cơ chế tiền xử lý dữ liệu huấn luyện bằng việc transfer từ một mô hình chung được đào tạo từ một lượng lớn các dữ liệu không được gán nhãn.
* Mô hình BERT được tạo ra để giải quyết sự thiếu hụt dữ liệu có nhãn chất lượng cao để đào tạo mô hình.

Trong khi các mô hình như Word2vec, FastText tìm ra 1 vector đại diện cho mỗi từ dựa trên một tập ngữ liệu lớn nên không thể hiện được sự đa dạng của ngữ cảnh, BERT mở rộng khả năng của các phương pháp trước đây bằng cách tạo các biểu diễn theo ngữ cảnh dựa trên các từ trước và sau đó để dẫn đến một mô hình ngôn ngữ với ngữ nghĩa phong phú hơn.

Vector đại diện được sinh ra từ mô hình BERT được tính chỉnh với các lớp đầu ra bổ sung đã tạo ra nhiều kiến trúc cải tiến đáng kể cho các nhiệm vụ xử lý ngôn ngữ tự nhiên như Question Answering, Language Inference,... mà không cần thay đổi quá nhiều từ các kiến trúc cũ.

**1.4.1.2: Kiến trúc của BERT**

Kiến trúc của mô hình BERT là một kiến trúc đa tầng gồm nhiều lớp Bidirectional Transformer encoder và sự phát hành trong thư viện tensor2tensor.

BERT có 2 mô hình:

* BERTbase​: L=12, H=768, A=12, Tổng tham số =110M
* BERTlarge​: L=24, H=1024, A=16, Tổng tham số =340M

Trong đó:

* L: số lớp Transformer (blocks) được sử dụng
* H: kích thước của các lớp ẩn
* A: số heads ở lớp attention, mỗi head sẽ thực hiện một self-attention

### 1.4.2. PhoBERT là gì?

PhoBERT là một phiên bản cải tiến của BERT, là một pre-trained được huấn luyện monolingual language (đơn ngữ), được huấn luyện dành riêng cho tiếng Việt.

Tên gọi “PhoBERT” được lấy cảm hứng từ “Phở”, một món ăn truyền thống của Việt Nam. Tương tự như BERT, PhoBERT cũng có 2 phiên bản là PhoBERTbase​ với 12 transformers block và PhoBERTlarge​ với 24 transformers block.

PhoBERT được train trên khoảng 20GB dữ liệu bao gồm 1GB kho dữ liệu Wikipedia Tiếng Việt (Vietnamese Wikipedia corpus) và 19GB còn lại lấy từ kho tin tức Việt (Vietnamese news corpus).

### 1.4.3. Cách thức PhoBERT hoạt động

Đầu tiên, PhoBERT sử dụng RDRSegmenter của VnCoreNLP để tách từ cho dữ liệu đầu vào trước khi qua BPE encoder.

* VnCoreNLP là một package NLP trong Tiếng Việt, hỗ trợ tokenize (tách từ) và các tác vụ NLP khác.
* RDRSegmenter là công cụ sử dụng để phân đoạn từ tiếng Việt một cách nhanh và chính xác.
* BPE encoder (Byte Pair Encoding) kỹ thuật nén dữ liệu hoạt động bằng cách thay thế các cặp byte liên tiếp có tần suất lớn bằng một byte không tồn tại trong dữ liệu.
* Cụ thể, 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. Quá trình gồm 5 bước:
  1. Khởi tạo từ điển (vocabulary).
  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).
  3. Thống kê tần suất xuất hiện theo cặp của toàn bộ token trong từ điển.
  4. Gộp các cặp có tần suất xuất hiện lớn nhất để tạo thành một n-gram theo level character mới cho từ điển.
  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.

Sử dụng fastBPE để phân đoạn những câu với các đơn vị từ khóa phụ. FastBPE là package hỗ trợ tokenize từ (word) thành các từ phụ (subwords) theo phương pháp mới nhất được áp dụng cho các pretrain model NLP hiện đại như BERT và các biến thể của BERT.

PhoBERT sử dụng task Masked Language Model để huấn luyện, bỏ task Next Sentence Prediction.

* Masked Language Model: là một loại học tự giám sát trong đó mô hình học cách tạo ra văn bản mà không có nhãn hoặc chú thích rõ ràng, nó lấy sự giám sát của mình từ văn bản đến. Masked Language Model có thể được sử dụng để thực hiện nhiều tác vụ NLP khác nhau như phân loại văn bản, trả lời câu hỏi và tạo văn bản.
* Next Sentence Prediction: cơ chế huấn luyện yêu cầu mô hình phải dự đoán xem hai câu bất kỳ có phải là hai câu liên tiếp trong một văn bản hay không.

## 1.5 Mô hình BiLSTM

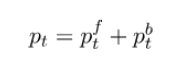
### 1.5.1. Định nghĩa

LSTM hai chiều hoặc BiLSTM (Bidirectional Long Short-Term Memory) là thuật ngữ được sử dụng cho mô hình trình tự chứa hai lớp LSTM , một lớp để xử lý đầu vào theo hướng thuận và lớp kia để xử lý theo hướng ngược. Nó thường được sử dụng trong các nhiệm vụ liên quan đến NLP. Trực giác đằng sau cách tiếp cận này là bằng cách xử lý dữ liệu theo cả hai hướng, mô hình có thể hiểu rõ hơn mối quan hệ giữa các chuỗi (ví dụ: biết các từ sau và các từ trước trong câu).

Để hiểu rõ hơn điều này chúng ta hãy xem một ví dụ. Câu đầu tiên là “Cô ấy có một chiếc xe mới” và câu thứ hai là “Cô ấy có thể đến muộn”. Trong cả hai câu này, từ “có” có ý nghĩa khác nhau và mối quan hệ này phụ thuộc vào các từ sau và trước trong câu lệnh. LSTM hai chiều giúp máy hiểu mối quan hệ này tốt hơn so với LSTM một chiều. Khả năng này của BiLSTM làm cho nó trở thành kiến ​​trúc phù hợp cho các nhiệm vụ như phân tích tình cảm , phân loại văn bản và dịch máy.

### 1.5.2. Kiến trúc lớp BiLSTM

Kiến trúc của LSTM hai chiều bao gồm hai LSTM một chiều xử lý chuỗi theo cả hai hướng tiến và lùi. Kiến trúc này có thể được hiểu là có hai mạng LSTM riêng biệt, một mạng lấy chuỗi mã thông báo như cũ trong khi mạng kia nhận theo thứ tự ngược lại. Cả hai mạng LSTM này đều trả về một vectơ xác suất làm đầu ra và đầu ra cuối cùng là sự kết hợp của cả hai xác suất này. Nó có thể được biểu diễn dưới dạng:

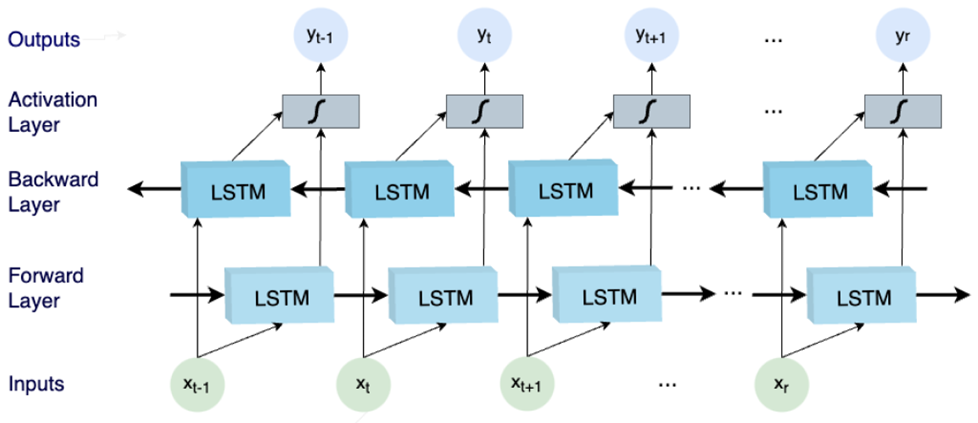


Trong đó:

 : Véc tơ xác suất cuối cùng của mạng

: Véc tơ xác suất từ mạng LSTM xuôi

: Véc tơ xác suất từ mạng LSTM ngược



#### Hình 1.2 Kiến trúc lớp BiLSTM

Cấu trúc cơ bản:

* Forward Layer: Lớp này xử lý dữ liệu theo thứ tự từ trước đến sau. Nhận đầu vào Xt tại thời điểm t và tạo ra đầu ra Yt
* Backward Layer: Lớp này xử lý dữ liệu theo thứ tự ngược lại, từ sau về trước. Nhận đầu vào Xt+1 và tạo ra đầu ra Yt+1

LSTM Cells: Mỗi lớp Forward và Backward đều chứa cá ô LSTm giúp mạng ghi nhớ thông tin trong khoảng thời gian dài. Các ô này có khả năng duy trì và quên thông tin một cách linh hoạt, điều này rất quan trọng trong việc xử lý dữ liệu tuần tự

Kết hợp đầu ra: Đầu ra từ cả 2 lớp Forward và Backward được kết hợp lại. Cả hai mạng LSTM này đều trả về một vectơ xác suất làm đầu ra và đầu ra cuối cùng là sự kết hợp của cả hai xác suất này

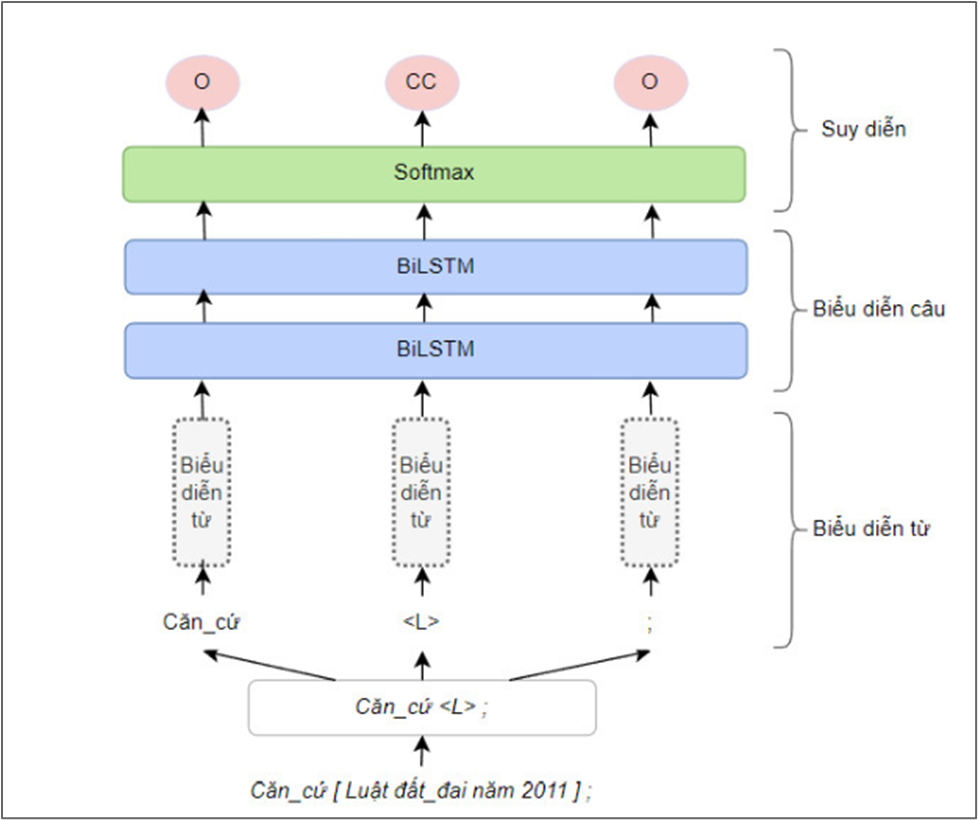
Activation Layer (Lớp kích hoạt): Sau khi kết hợp dầu ra từ cá lớp LSTM, lớp kích hoạt sẽ được sử dụng để chuyển đổi đầu ra thành định dạng mong muốn, ở đâu là xác suất cho các lớp phân loại

Ví dụ:

Đầu vào: Cho câu đầu vào “Căn\_cứ Luật đất\_đai năm 2011 ;”.

Xử lý câu đầu vào thành chuỗi: “Căn\_cứ ;”, trong đó đại diện cho loại thực thể “luật”, thay thế cho thực thể “Luật đất\_đai năm 2011”.

Đầu ra: Chuỗi nhãn đầu ra tương ứng: “O, CC, O”



#### Hình 1.3 Mô hình BiLSTM cho phân loại quan hệ giữa các thực thể

## 1.6. Kết luận

Chương này giới thiệu những kiến thức cơ bản và quan trọng về học máy, đặc biệt tập trung vào các khái niệm nền tảng như học có giám sát, học không giám sát và học tăng cường.

Giới thiệu khái quát các thuật toán học máy phổ biến, từ những thuật toán truyền thống như hồi quy tuyến tính, cây quyết định đến các thuật toán hiện đại như mạng nơ-ron nhân tạo. Ngoài ra, chương còn đi sâu vào các kiến trúc mạng nơ-ron sâu như CNN và RNN, cũng như các mô hình ngôn ngữ lớn như BERT và PhoBERT. Việc nắm vững những kiến thức này là nền tảng quan trọng để tiếp cận và ứng dụng học máy vào các bài toán thực tế.

# Chương 2. Thử nghiệm

## 2.1 Bộ dữ liệu

### 2.1.1 Tổng quan dữ liệu

Bộ dữ liệu này được thu thập từ các bài báo của các trang báo mạng uy tín như là VnExpress, Báo Thanh Niên, VTV24, VietNamNet,... với các chủ đề về thể thao, giáo dục, giải trí, xã hội, kinh tế. Bộ dữ liệu bao gồm các thuộc tính sau:

* Câu văn: (văn bản tiếng việt) Câu văn được thu thập từ các bài báo đáp ứng đúng định dạng của 1 câu trong tiếng Việt
* Nhãn cảm xúc: (số) dưới dạng 0/1/2 với
  + 0: Các câu mang cảm xúc tiêu cực
  + 1: Các câu mang cảm xúc bình thường
  + 2: Các câu mang cảm xúc tích cực
* Ví dụ về dữ liệu:

#### Bảng 2.1. Dữ liệu thí dụ

|  |  |
| --- | --- |
| **Câu văn** | **Nhãn cảm xúc** |
| Nếu như vậy, tôi sẽ chỉ trích HLV rất nhiều, và tôi không thể làm điều đó. | 0 |
| Sang hiệp hai, thẻ đỏ chỉ sau sáu phút của Hồng Quân khiến Bình Định chỉ còn biết thủ thế chống đỡ. | 0 |
| Nhưng Guardiola chưa quyết định tương lai không phải vì chờ kết quả xét xử Man City. | 1 |
| Công đoàn đã làm tốt công tác thăm hỏi những gia đình chị em ốm đau hoặc gặp chuyện vui buồn. | 2 |
| ………………. | ……… |
| Phim nhận đánh giá tích cực từ giới phê bình. | 2 |
| Ngoài ra nếu có thêm thắc mắc, học viên trao đổi ngoài giờ đều được bà tận tình hướng dẫn. | 2 |
| Vụ việc khiến 2 người trên xe thương vong. | 0 |
| Nhờ mạng xã hội , Xuân Huy và Xuân Hùng có thêm người bạn cùng cảnh ngộ . | 1 |

### 2.1.2 Các tập dữ liệu

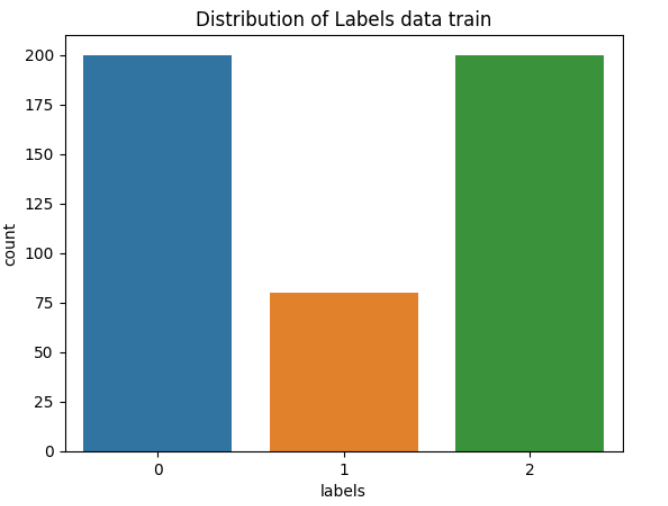
Tổng số dữ liệu tìm được là 600 câu trong đó:

* Số câu mang cảm xúc tiêu cực:249 câu
* Số câu mang cảm xúc bình thường:106 câu
* Số câu mang cảm xúc tích cực**:** 245 câu

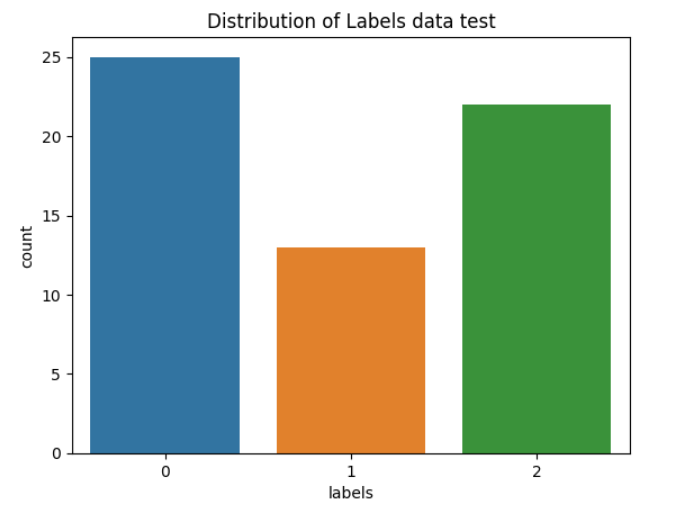
Dữ liệu được chia ra làm 3 tập chính bao gồm:

* **Tập train**: (Chiếm 80% - 480 câu) Sử dụng để huấn luyện mô hình, giúp nó học các đặc trưng từ dữ liệu.
* **Tập valid**: (Chiếm 10% - 60 câu) Dùng để điều chỉnh các tham số của mô hình và kiểm tra hiệu suất trong quá trình huấn luyện, giúp tránh hiện tượng overfitting.
* **Tập test**: (Chiếm 10% - 60 câu) Sử dụng để đánh giá mô hình sau khi đã huấn luyện, giúp kiểm tra khả năng tổng quát của mô hình với dữ liệu chưa thấy.

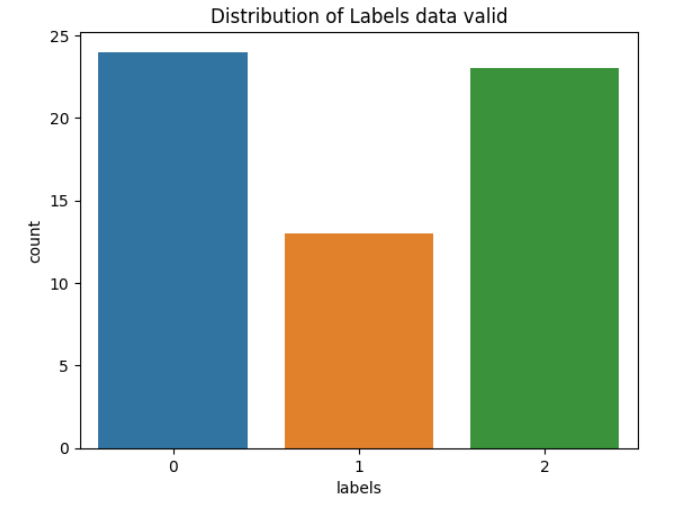
## 2.3 Các phân tích thống kê về bộ dữ liệu và đồ họa



#### Hình 2.1. Biểu đồ tỉ lệ cảm xúc các câu trong bộ dữ liệu train

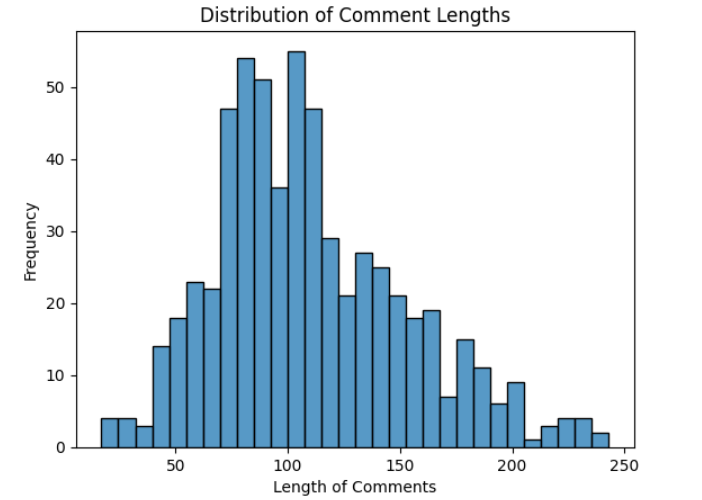


#### Hình 2.2. Biểu đồ tỉ lệ cảm xúc các câu trong bộ dữ liệu test



#### Hình 2.3. Biểu đồ tỉ lệ cảm xúc các câu trong bộ dữ liệu valid

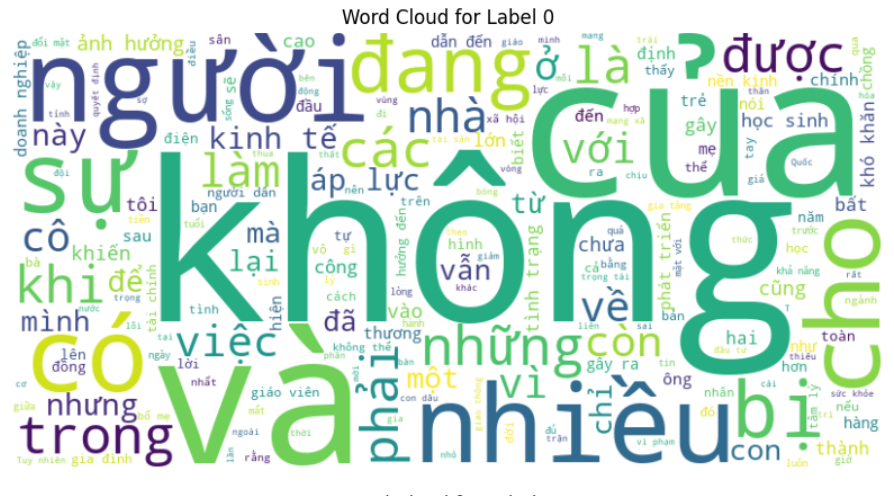
Tỉ lệ giữa các nhãn trong các bộ data được giữ ở mức gần tương đương nhau: **Nhãn 0** (40% - 45%), **Nhãn 1** (10% - 20%), **Nhãn 2 (**40% - 45%)



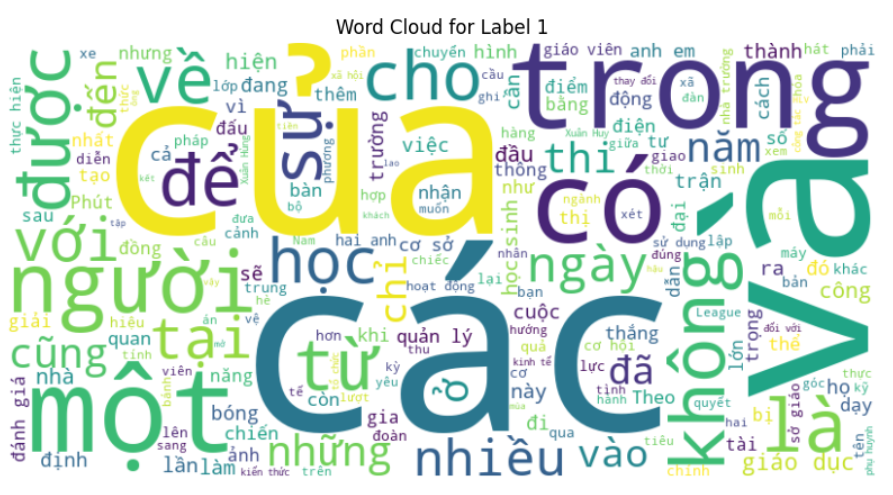
#### Hình 2.4 Biểu đồ thể hiện phân bố độ dài các câu

Tập dữ liệu chứa 600 bình luận với độ dài trung bình khá đa dạng. Sự phân tán về độ dài bình luận cho thấy có nhiều bình luận ngắn gọn, nhưng cũng có những bình luận dài và chi tiết. Điều này cho thấy sự phong phú trong cách diễn đạt ý kiến của người dùng.

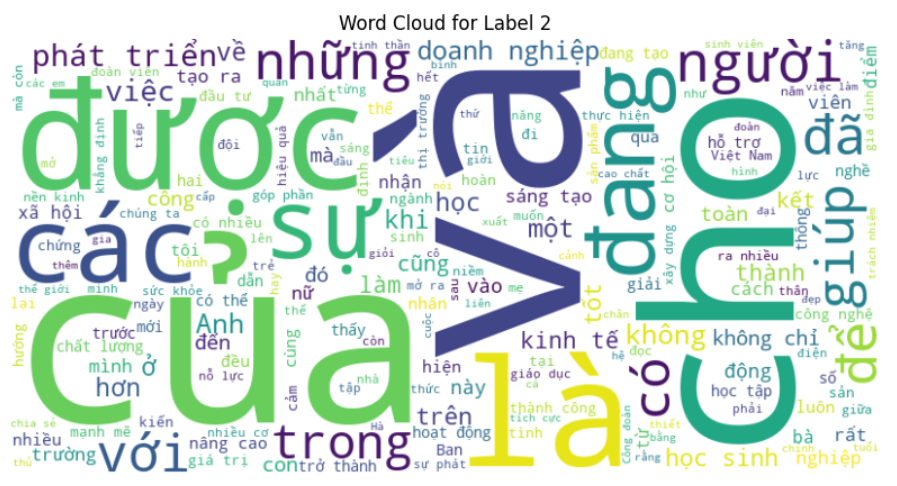
* Số lượng bình luận trong tập dữ liệu: 600
* Trung bình độ dài bình luận: 109,45
* Độ lệch chuẩn: 42,55
* Độ dài bình luận ngắn nhất: 17
* 25% bình luận có độ dài ngắn hơn giá trị này: 79,75
* Độ dài bình luận trung bình: 102
* 75% bình luận có độ dài ngắn hơn giá trị này: 136
* Độ dài bình luận dài nhất: 242



#### Hình 2.5 Biểu đồ thể hiện tần suất từ xuất hiện trong nhãn tiêu cực



#### Hình 2.6 Biểu đồ thể hiện tần suất từ xuất hiện trong nhãn bình thường



#### Hình 2.7 Biểu đồ thể hiện tần suất từ xuất hiện trong nhãn tích cực

Phân tích các từ phổ biến trong từng nhãn cung cấp cái nhìn sâu sắc về các chủ đề chính và cảm xúc của tác giả bài báo đặt vào câu văn. Việc xác định những từ khóa này giúp có cái nhìn cụ thể về đặc trưng của từng nhãn cảm xúc. Dưới đây là đánh giá tổng quan về từng nhãn dựa trên top 10 từ phổ biến:

* **Nhãn 0**: Các từ như "người", "không", và "nhiều" chỉ ra sự tập trung vào các vấn đề cá nhân hoặc xã hội tiêu cực, cho thấy người dùng có thể đang đối mặt với khó khăn hoặc cảm xúc tiêu cực.
* **Nhãn 1**: Sự xuất hiện của "và", "của", và "các" thường thể hiện những câu mang tính chất trung tính, đơn thuần là cung cấp thông tin và không thể hiện cảm xúc.
* **Nhãn 2**: Các từ như "cho", "được", và "là" cho thấy sự nhấn mạnh vào sự tốt đẹp, vui vẻ từ các chủ đề từ đó đem lại cảm xúc tích cực cho câu văn

## 2.3 Mô hình

### 2.3.1 Tiền xử lý:

#### 2.3.1.1: PhoBert

**Bước 1**: Sử dụng RDRSegmenter từ VnCoreNLP để phân đoạn từ trong tiếng Việt, xử lý để đảm bảo rằng mỗi từ được tách chính xác và phù hợp với ngữ nghĩa tiếng Việt (do trong tiếng Việt không có dấu rõ rang cho các từ ghép để máy hiểu).

Ví dụ: Đưa đầu vào là một câu, câu này sẽ được tách thành các từ có nghĩa.

* Đầu vào: “Học viện Công nghệ Bưu chính Viễn thông đón đầu xu hướng công nghệ cao.”
* Đầu ra: [['Học\_viện', 'Công\_nghệ', 'Bưu\_chính', 'Viễn\_thông', 'đón\_đầu', 'xu\_hướng', 'công\_nghệ\_cao', '.']]

**Bước 2**: Kết hợp từ đã phân đoạn lại thành câu, sau đó thêm vào danh sách câu.

* Đầu vào: [['Học\_viện', 'Công\_nghệ', 'Bưu\_chính', 'Viễn\_thông', 'đón\_đầu', 'xu\_hướng', 'công\_nghệ\_cao', '.']]
* Đầu ra: “Học\_viện Công\_nghệ Bưu\_chính Viễn\_thông đón\_đầu xu\_hướng công\_nghệ\_cao.”

**Bước 3**: Thêm nhãn tương ứng vào danh sách nhãn.

* Mỗi câu trong tập dữ liệu sẽ có các nhãn tương ứng, câu nào không có nhãn sẽ được loại bỏ, nhãn này sẽ được đưa vào danh sách nhãn một cách chính xác.

**Bước 4**: Mã hóa và chuẩn hóa chuỗi cho các tập train, test, valid.

* Mã hóa câu của tập dưới dạng BPE.
* Thêm token đầu và cuối chuỗi: '<s> ' + (câu đã mã hóa) + ' </s>'
* Mã hóa câu trước đó bằng cách chuyển từng từ trong chuỗi thành một ID (int) dựa vào từ điển của mô hình, nếu từ không có trong từ điển, nó sẽ được xử lý dựa trên các quy tắc mã hóa, như chia thành các byte-pair (BPE) nhỏ hơn.
* Ví dụ: Đầu vào: '<s> ' + 'Mai mà nóng thì ở nhà không đi học đâu!' + ' </s>'.

Đầu ra: Câu sẽ được mã hóa thành các id: [0,2454,64,898,54,25,69,17,57,222,64002], trong đó: 0 và 60042 đại diện cho <s> và </s>, các id còn lại tương ứng với các từ trong câu, riêng từ “đâu!” không được mã hóa do dấu chấm than không được tách riêng.

**Bước 5**: Chuẩn hóa độ dài của các ID chuỗi về độ dài cố định.

* Đặt độ dài tối đa là 150.
* Cắt hoặc thêm giá trị 0 vào các chuỗi sau khi mã hóa cho đến khi đạt độ dài 150.

**Bước 6**: Tạo masks cho dữ liệu đầu vào.

* Phân loại vị trí của các token thực (token thực sự mang thông tin) và các token padding (token được thêm vào để đảm bảo độ dài chuỗi cố định).

#### 2.3.1.2: BiLSTM:

Tiền xử lý là một bước quan trọng khi làm việc với mô hình BiLSTM trong phân tích cảm xúc, vì nó giúp cải thiện hiệu suất của mô hình bằng cách làm sạch và chuẩn bị dữ liệu văn bản. Các bước tiền xử lý:

1. Làm sạch văn bản:

* Chuyển sang chữ thường: Chuyển toàn bộ văn bản thành chữ thường để tránh phân biệt từ viết hoa và viết thường (ví dụ: “Tốt” và “tốt”).
* Loại bỏ nhiễu: Loại bỏ các ký tự đặc biệt, dấu câu và các ký tự không thuộc ASCII không đóng góp ý nghĩa cảm xúc.

Ví dụ:

* Văn bản ban đầu: “Món ăn thật tuyệt :))”
* Sau khi loại bỏ: “Món ăn thật tuyệt”
* Loại bỏ từ dừng: Loại bỏ các từ thông dụng như “có”, “là”, “và” không mang nhiều thông tin về cảm xúc. Tuy nhiên, bước này có thể tùy chọn nếu từ dừng có giá trị cảm xúc trong dữ liệu.

Ví dụ:

* Văn bản ban đầu: “Món ăn rất ngon và dịch vụ cũng tốt”
* Sau khi loại bỏ từ dừng: “Món ăn ngon dịch vụ tốt”
* Lemmatization: Chuyển các từ về dạng gốc hoặc dạng cơ sở của chúng.

Ví dụ:

* Văn bản ban đầu: “Những con mèo chạy nhanh hơn những con chó.”
* Sau khi lemmatization: “Con mèo chạy nhanh hơn con chó.”

1. Phân tách từ (Tokenization)

* Phân tách từ:Tách văn bản thành các từ (token) riêng lẻ, cho phép mô hình BiLSTM xử lý từng từ một cách độc lập.

Ví dụ:

* Văn bản ban đầu: “Tôi thích đi dạo vào buổi chiều.”
* Sau khi phân tách: [“Tôi”, “thích”, “đi”, “dạo”, “vào”, “buổi”, “chiều”]
* Phân tách từ con (Sub-word Tokenization): Đối với một số tác vụ, tách từ thành các từ con (ví dụ: “không hài lòng” thành “không”, “hài”, “lòng”) có thể hiệu quả, đặc biệt khi xử lý các từ hiếm hoặc không có trong từ vựng.

1. Padding và Truncating (Đệm thêm hoặc cắt bớt):

* Padding: Do mô hình BiLSTM cần các chuỗi đầu vào có độ dài nhất định, nên thêm padding để làm cho chuỗi có cùng độ dài.

Ví dụ:

* Độ dài tối đa đặt là 125
* Văn bản ban đầu sau khi mã hóa: [1, 234, 56] (tương ứng với câu ngắn "món ăn ngon")
* Sau khi padding để đạt độ dài 125: [0, 0, 0, ..., 1, 234, 56] (các giá trị 0 sẽ được thêm vào đầu chuỗi đến khi đạt đủ độ dài 125)
* Truncating: Đối với những chuỗi dài hơn độ dài tối đa, cắt bớt chúng để tránh vượt quá giới hạn đầu vào của mô hình.

Ví dụ:

* Độ dài tối đa đặt là 125
* Văn bản ban đầu sau khi mã hóa: [3, 432, 76, 22, 9, 15, ..., 255, 102] (tương ứng với câu dài như "dịch vụ này rất đặc biệt và đồ ăn được chế biến hoàn hảo, với bầu không khí tuyệt vời và đội ngũ nhân viên thân thiện, tạo nên một trải nghiệm khó quên")
* Sau khi cắt: [3, 432, 76, ..., 255, 102] (giữ nguyên 125 phần tử đầu tiên và loại bỏ phần còn lại nếu vượt quá)

1. Chuyển đổi văn bản thành chuỗi số:

* Encode: Sử dụng một từ vựng để chuyển các từ thành các mã số. Có thể tạo từ vựng từ dữ liệu huấn luyện hoặc sử dụng một tokenizer được huấn luyện sẵn (ví dụ: từ embeddings của GloVe, Word2Vec hoặc FastText).

Ví dụ:

* Giả sử từ vựng: {"Tôi": 1, "thích": 2, "đồ": 3, "ăn": 4, “và”: 5, “uống”:6}
* Văn bản: “Tôi thích đồ ăn và đồ uống”
* Sau khi mã hóa: [1, 2, 3, 4, 5, 3, 6]
* Embedding Layer Preparation: Nếu sử dụng các embeddings đã được huấn luyện trước, ánh xạ các từ vào các vector embeddings mà BiLSTM sẽ sử dụng làm đầu vào.

Ví dụ:

* Nếu sử dụng embeddings từ GloVe, từ “ngon” sẽ được ánh xạ thành một vector đặc trưng ("ngon" → [0.2, 0.7, -0.1, 0.4]).

1. Xử lý dữ liệu mất cân bằng (tùy chọn):

Nếu dữ liệu phân tích cảm xúc không cân bằng (ví dụ: nhiều đánh giá tích cực hơn tiêu cực), có thể cân nhắc các kỹ thuật lấy mẫu lại hoặc thêm trọng số cho các lớp để tránh dự đoán bị thiên lệch.

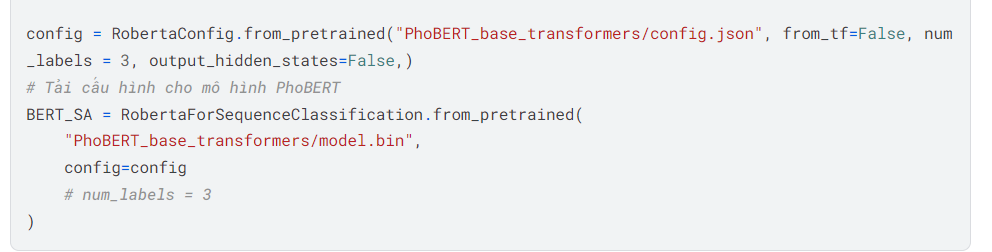
Nếu tập dữ liệu có 80% phản hồi tích cực và 20% phản hồi tiêu cực: Có thể thực hiện over-sampling để tăng số lượng phản hồi tiêu cực, giúp mô hình không bị thiên lệch về phản hồi tích cực

### 2.3.2 Quy trình xây dựng mô hình:

Xác định biến đầu vào và nhãn:

* Biến đầu vào (X): Là câu văn được trích trong văn bản lấy từ các báo chính thống.
* Biến nhãn (y): Là thái độ của câu văn đó, 0 - mang giá trị tiêu cực, 1 - mang giá trị bình thường, 2 - mang giá trị tiêu cực

### 2.3.3 Xây dựng mô hình phoBERT.



#### Hình 2.8 Nạp mô hình phoBERT

Để xây dựng mô hình, cần import những thư viện cần thiết để tải và config mô hình: RobertaForSequenceClassification, RobertaConfig, AdamW. Nhóm sử dụng pretrained\_model để tiết kiệm thời gian và nâng cao hiệu suất. Một số cấu hình cài đặt:

* from\_tf=False: Chỉ định rằng cấu hình không được chuyển đổi từ một mô hình TensorFlow (mà là từ PyTorch).
* num\_labels=3: Số lượng nhãn là 3
* output\_hidden\_states=False: Không trả về các trạng thái ẩn từ các lớp trong mô hình.

Nhóm sử dụng PhoBERTbase được tinh chỉnh theo yêu cầu cụ thể của bài toán đang giải quyết, ở đây là phân loại cảm xúc cho văn bản. PhoBERT, giống như BERT và RoBERTa, có kiến trúc gồm nhiều lớp transformer lặp lại, mỗi lớp bao gồm các thành phần như Multi-Head Self-Attention, Layer Normalization, và Feed-Forward Neural Network.

### Một số thuộc tính sử dụng trong quá trình training:

- **Epoch**: là một thuật ngữ chỉ một vòng lặp mà mô hình sẽ duyệt toàn bộ tập dữ liệu huấn luyện để cập nhật trọng số của nó, mục tiêu là tìm ra trọng số tốt nhất cho mô hình. Một epoch được tính khi đưa được tất cả dữ liệu trong tập train vào mạng neural network

- **batch\_size**: là số lượng mẫu dữ liệu trong một lần huấn luyện. VD: batch\_size=32, tức là ta cho ngẫu nhiên 32 câu chạy lan truyền đến mạng NN, tiếp theo ta đưa tiếp 32 câu ngẫu nhiên k lặp với câu trước đó vào mạng, cứ như vậy đến khi không còn câu nào nữa thì hoàn thành 1 epoch.

- **AdamW:** Một phiên bản của thuật toán tối ưu hóa Adam, sử dụng weight decay để tránh overfitting.

o **Weight-decay:** 1 kỹ thuật để tránh overfitting, đó là làm giảm trọng số. Trong quá trình huấn luyện có thể dẫn đến trọng số quá lớn, khiến cho mô hình ghi nhớ dữ liệu huấn luyện hơn là học tổng quát hoá.

- **get\_linear\_schedule\_with\_warmup:** Hàm để tạo lịch trình học (learning rate schedule) cho việc điều chỉnh learning rate trong quá trình huấn luyện.

- **optimizer\_grouped\_parameters:** Tạo nhóm các tham số với weight decay cho các tham số cụ thể.

- **Scheduler:** Tạo một lịch trình học cho việc điều chỉnh learning rate, bao gồm giai đoạn "warm-up".

- **Dropout:** 1 kỹ thuật được sử dụng trong mạng NN để tránh overfitting trên tập huấn luyện bằng cách loại bỏ các neural với xác suất p>0

- **loss.backward():** hàmtính toán gradient loss

* **hàm mất mát (loss function):** thể hiện sự chênh lệch giữa 2 đại lượng là label được dự đoán và label đúng. Loss càng nhỏ thì model càng tốt

- **torch.nn.utils.clip\_grad\_norm**\_: Giới hạn gradient để tránh vấn đề "exploding gradients".

* **Gradient:** độ dốc, biểu diễn tốc độ thay đổi của hàm. Hiểu nôm na, ứng dụng của nó là để tối ưu các trọng số trong mạng bằng cách tìm cực đại, cực tiểu của hàm.
* **Exploding gradient:** Đây là hiện tượng gradient quá lớn do tích tụ gradient ở những lớp cuối đặc biệt hay xảy ra đối với câu dài

- **optimizer.step():** cập nhật trọng số của mô hình

- **scheduler.step():** cập nhật learning rate

o **learning rate:** Tốc độ học (learning rate), thường được kí hiệu là α*α* hoặc đôi khi là η*η*, cho biết mức độ thay đổi của các trọng số sau mỗi lần được cập nhật. Nó có thể được cố định hoặc thay đổi thích ứng. Phương thức phổ biến nhất hiện nay là Adam, đây là phương thức thích nghi với tốc độ học.

- **avg\_train\_loss**: tính toán trung bình loss của toàn hệ thống

#### Bảng 2.2: Thiết lập tham số huấn luyện

|  |  |  |
| --- | --- | --- |
| **Tham số** | **Giá trị** | **Mô tả** |
| epochs | 15 | số lượng epoch huấn luyện |
| batch\_size | 32 | số lượng mẫu dữ liệu được xử lý trong một lần chạy của mô hình |
| MAX\_LEN | 150 | Đặt độ dài tối đa cho mỗi chuỗi sau khi được chuẩn hóa |
| config | RobertaConfig.from\_pretrained("PhoBERT\_base\_transformers/config.json", from\_tf=False, num\_labels = 3, output\_hidden\_states=False,) | cấu hình cho mô hình PhoBert |
| device | ‘cuda’ | sử dụng GPU |
| optimizer | AdamW(optimizer\_grouped\_parameters, lr=3e-5) | tốc độ học ban đầu |
| scheduler | get\_linear\_schedule\_with\_warmup(optimizer, num\_warmup\_steps=0, num\_training\_steps=len(train\_dataloader) \* epochs) | điều chỉnh tốc độ học theo thời gian |
| best\_eval | best\_eval = (eval\_accuracy / nb\_eval\_steps) | biến lưu giá trị đánh giá tốt nhất |
| BERT\_SA.save\_pretrained(f'bert\_pretrain/') |  | lưu mô hình tốt nhất |
| avg\_train\_loss | total\_loss / len(train\_dataloader) | Tính toán loss trung bình cho epoch |

### 2.3.4 Xây dựng mô hình BiLSTM



#### Hình 2.9 Mô tả cho mô hình BiLSTM

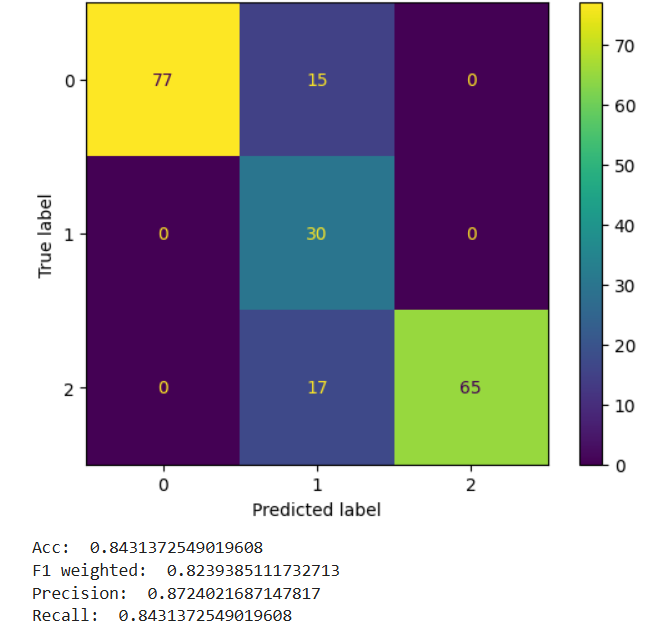
Mô hình BiLSTM nhóm sử dụng không có quá nhiều thay đổi, tuy nhiên cũng có một số điểm nhóm cải tiến để tăng hiệu quả cho mô hình:

* Embedding: Mô hình này sử dụng một lớp embedding để chuyển đổi từ ngữ thành các vector nhúng. Điều này giúp mô hình có khả năng hiểu được ngữ nghĩa của từ ngữ và cải thiện hiệu suất của mô hình.
* Dropout: Mô hình này sử dụng dropout, một kỹ thuật chống lại việc quá khớp (overfitting) trong mạng nơ-ron. Dropout hoạt động bằng cách tắt ngẫu nhiên một số nơ-ron trong quá trình huấn luyện, giúp mô hình trở nên mạnh mẽ hơn và ít phụ thuộc vào một số đặc trưng cụ thể.
* Projection Layer: Mô hình này sử dụng một lớp dự đoán (projection layer) để chuyển đổi đầu ra của LSTM hai chiều thành dự đoán cho mỗi lớp đầu ra. Điều này giúp mô hình có thể được sử dụng cho nhiều tác vụ khác nhau, không chỉ giới hạn ở tác vụ phân loại.

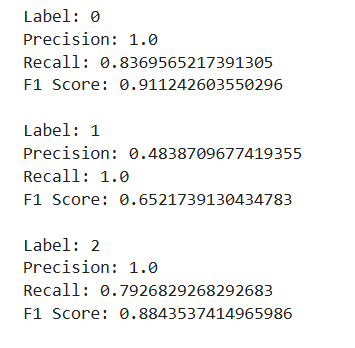
### 2.3.5 Kết Quả Cuối Cùng

Cuối cùng, sau khi hoàn thành các bước trên, chúng tôi đã có được một mô hình nhận diện cảm xúc các câu xuất hiện trong các bài báo chính thống. Các kết quả thu được từ hai mô hình sẽ được so sánh trực quan để có cái nhìn sâu sắc hơn về khả năng dự đoán của cả hai.

### Mô hình phoBERT

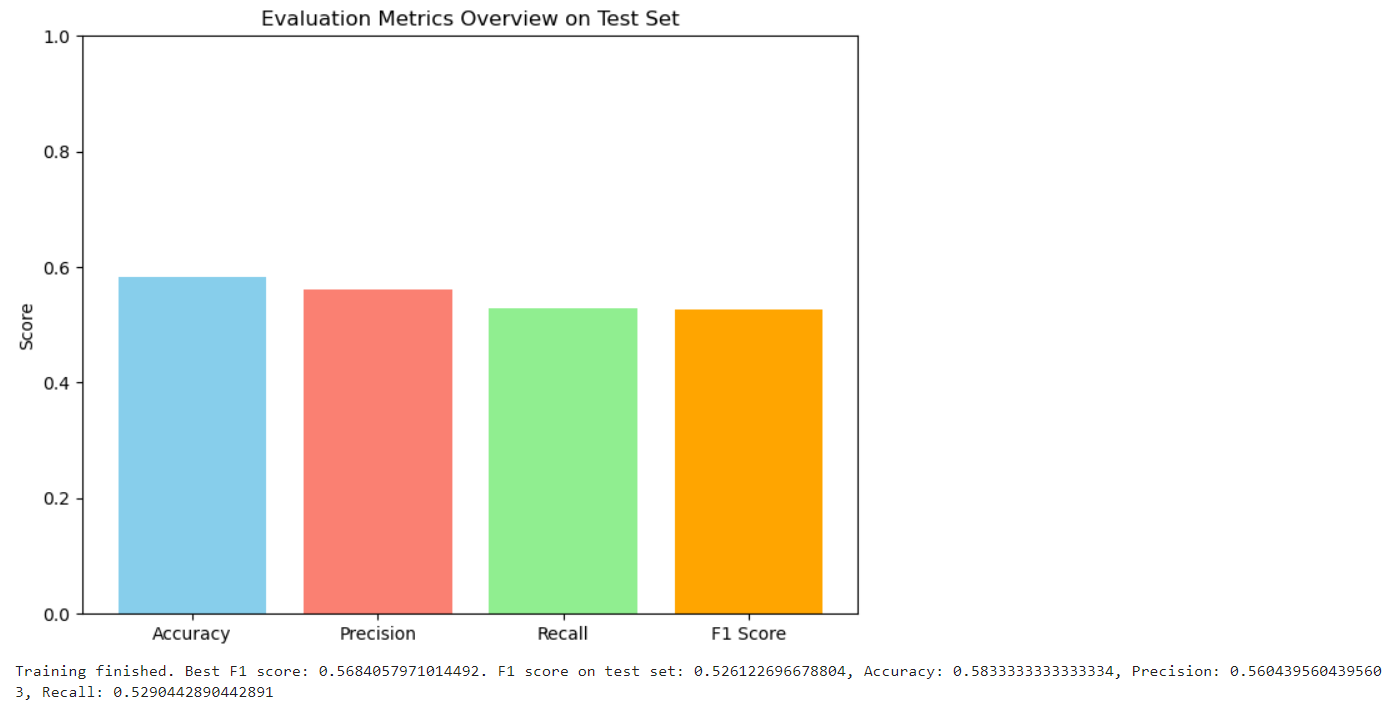


#### Hình 2.10 Kết quả đánh giá độ chính xác của mô hình phoBERT

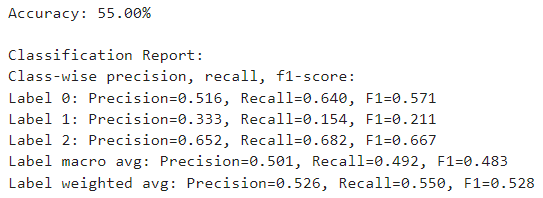


#### Hình 2.11 Kết quả dự đoán đúng của từng nhãn

### Mô hình BiLSTM



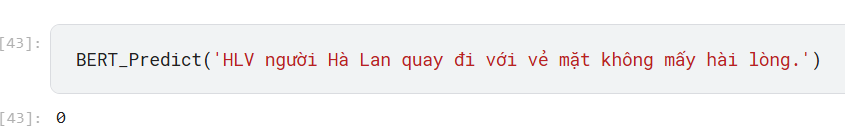
#### Hình 2.12 Kết quả dự đoán với mô hình BiLSTM

******

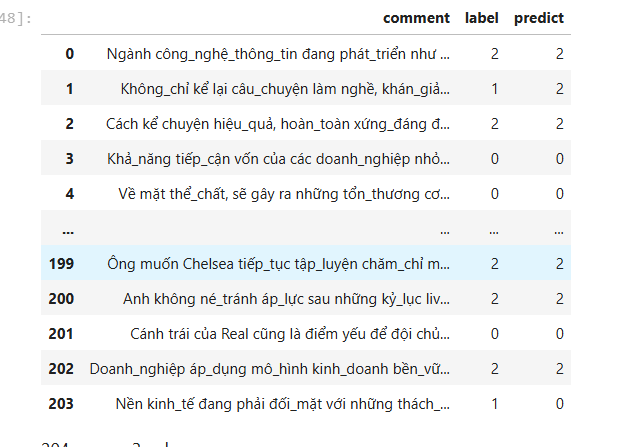
#### Hình 2.13 Kết quả dự đoán đối với từng nhãn của mô hình BiLSTM

### Kiểm tra hai mô hình với các câu mới

Mô hình PhoBERT:

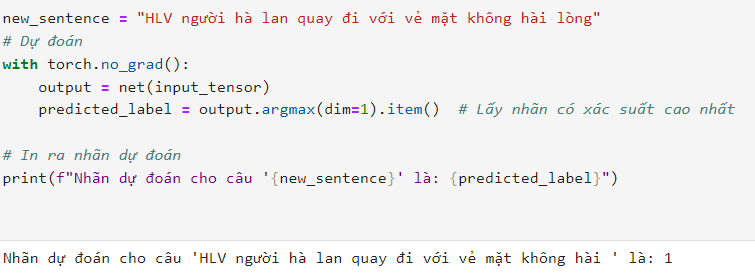


#### Hình 2.14 Kết quả dự đoán câu mấu của PhoBERT



#### Hình 2.15 Kết quả dự đoán tập test của PhoBERT

Mô hình BiLSTM



#### Hình 2.16 Kết quả dự đoán câu mấu của BiLSTM



#### Hình 2.17 Kết quả dự đoán tập test của BiLSTM

## 2.4 Kết luận

### So sánh kết quả

#### Bảng 2.3 So sánh kết quả hai mô hình

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Precision** | **Recall** | **F1-Score** |
| PhoBERT | 0.87 | 0.84 | 0.82 |
| BiLSTM | 0,46 | 0.46 | 0.54 |

### So sánh kích thước model

#### Bảng 2.4. So sánh kích thước hai mô hình

|  |  |
| --- | --- |
| **Model** | **Size (MB)** |
| PhoBERT | 540.03 |
| BiLSTM | 18.44 |

### Đánh giá

Sau quá trình train và thử nghiệm với bộ dữ liệu test. Nhóm nhận thấy các mô hình đều hoạt động khá tốt khi dự đoán các dữ liệu là các câu bình luận mới.

So sánh giữa hai mô hình đã huấn luyện, kết quả cho thấy PhoBert tốt hơn BiLSTM dựa trên yêu cầu bài toán là cần xác định văn bản là tích cực hay tiêu cực, do đó mô hình có chỉ số precision càng cao sẽ càng mang lại kết quả chính xác.

Nhận thấy, kích thước của model sử dụng PhoBert có sự chênh lệch lớn so với mô hình còn lại, điều này là do số lượng tham số cũng như số lượng lớp và chiều không gian của nó lớn hơn rất nhiều so với hai model còn lại. Ngoài ra PhoBert đã được pre-trained trên một lượng lớn văn bản Tiếng Việt, do đó model thường rất lớn vì chúng được thiết kế để có khả năng tổng quát hóa và nắm bắt được ngữ nghĩa phức tạp từ ngữ liệu lớn câu trên 3 tập.

# 

# Chương 3: Kết luận

## 1. Đã làm được

Trong nghiên cứu này, chúng tôi đã xây dựng và triển khai hai mô hình PhoBERT và BiLSTM để thực hiện nhiệm vụ phân loại thái độ trong đoạn văn bản (tích cực, tiêu cực, hoặc trung tính). Cả hai mô hình đều được thiết kế và tinh chỉnh với mục tiêu tối ưu hóa hiệu suất phân loại cảm xúc trong văn bản tiếng Việt, dựa trên tập dữ liệu có gán nhãn.

Các bước tiến hành bao gồm:

* Tiền xử lý văn bản: Chúng tôi chuẩn hóa dữ liệu đầu vào bằng cách chuyển về chữ thường, loại bỏ các ký tự đặc biệt và xử lý token không xác định để đảm bảo tính đồng nhất trong dữ liệu.
* Sử dụng mô hình embedding PhoBERT, chúng tôi chuyển đổi mỗi từ thành vector ẩn, giúp mô hình hiểu được mối quan hệ ngữ nghĩa giữa các từ
* Huấn luyện mô hình: PhoBERT và BiLSTM được huấn luyện trên tập dữ liệu phân loại cảm xúc với các nhãn tích cực, tiêu cực và trung tínhCác cải thiện đã thực hiện với từng mô hình:

***Với mô hình phoBERT***

* Sử dụng AdamW thay cho Adam với yếu tố weight-decay giúp giữ trọng số mô hình không quá lớn
* **get\_linear\_schedule\_with\_warmup** giúp điều chỉnh tốc độ học, tức là lúc đầu mô hình sẽ sử dụng warm-up (tăng dần learning rate) sau đó giảm dần để ổn định
* **torch.nn.utils.clip\_grad\_norm\_** được sử dụng để giới hạn gradient, tránh trường hợp **exploding gradient** (gradient quá lớn, gây khó khăn cho việc cập nhật trọng số).

***Với mô hình BiLSTM***

* Điều chỉnh Hyperparameters (hidden layer, embedding, Batch size, learning rate)
* Tối ưu hóa cấu trúc mạng BiLSTM
* Đánh giá và điều chỉnh trên từng nhãn

**Kết quả mô hình:**

* Độ chính xác: Cả hai mô hình đều đạt độ chính xác ấn tượng, với PhoBERT đạt khoảng 87% và BiLSTM đạt khoảng 46%. Kết quả này cho thấy mô hình có khả năng phân loại tốt các loại thái độ khác nhau trong văn bản tiếng Việt.
* F1 Score: PhoBERT đạt F1 Score trung bình macro là 0.85, và BiLSTM đạt khoảng 0.54. Điều này cho thấy các mô hình không chỉ có độ chính xác cao mà còn duy trì sự cân bằng giữa các lớp cảm xúc.
* Confusion Matrix: Ma trận nhầm lẫn giúp chúng tôi phân tích rõ hơn về cách mô hình phân loại từng nhãn cảm xúc, đồng thời chỉ ra được những nhãn mà mô hình phân biệt tốt và những nhãn còn nhầm lẫn.

**Hạn chế của nghiên cứu:**

* Cần cải thiện tính tổng quát của mô hình để không bị overfitting.
* Dữ liệu có thể chưa đầy đủ; việc mở rộng thêm mẫu và đa dạng cảm xúc có thể cải thiện kết quả.

## 2. Phương hướng tương lai

**Mở rộng bộ dữ liệu**: Việc thu thập thêm dữ liệu từ các nguồn và ngữ cảnh khác nhau sẽ giúp mô hình tổng quát hóa tốt hơn, cải thiện khả năng phân loại trên các dạng văn bản đa dạng.

**So sánh với các thuật toán khác**: Ngoài PhoBERT và BiLSTM, chúng tôi dự định thử nghiệm thêm các mô hình khác như Transformer hoặc CNN, và tiến hành đánh giá để tìm ra mô hình tối ưu nhất.

**Tối ưu hóa các tham số**: Sử dụng Grid Search hoặc Random Search để điều chỉnh tham số, đồng thời áp dụng kiểm tra chéo (cross-validation) nhằm nâng cao độ tin cậy của mô hình.

**Xem xét thêm yếu tố ngữ cảnh**: Có thể bổ sung thêm thông tin ngữ cảnh như thông tin về tác giả hoặc thời gian viết, giúp phân tích thêm các yếu tố tác động đến thái độ trong văn bản.

**Phát triển công cụ hỗ trợ phân tích cảm xúc**: Dựa trên mô hình đã xây dựng, chúng tôi có thể phát triển ứng dụng hỗ trợ phân tích cảm xúc trong văn bản tiếng Việt, phục vụ nghiên cứu hoặc theo dõi ý kiến công chúng.

# Tài liệu tham khảo

1. Đỗ Trung Tuấn, Bài giảng và slide môn học Phân tích thiết kế Hệ thống thông minh, PTIT, 2024.
2. Huu-Thanh Duong, Tram-Anh Nguyen-Thi, Vinh Truong Hoang, Vietnamese Sentiment Analysis under Limited Training Data Based on Deep Neural Networks. Retrieved from: <https://www.hindawi.com/journals/complexity/2022/3188449>
3. Luong Luc Phan, Phuc Huynh Pham, Kim Thi-Thanh Nguyen, Tham Thi Nguyen, Sieu Khai Huynh, Luan Thanh Nguyen, Tin Van Huynh, Kiet Van Nguyen, A2SL: From Aspect-Based Sentiment Analysis to Social Listening System for Business Intelligence. Retrieved from: <https://arxiv.org/abs/2105.15079>
4. Machine Learning cơ bản. (n.d.). Retrieved from https://machinelearningcoban.com/2016/12/26/introduce/
5. Vu, T. , Phân tích Khám phá Dữ liệu - EDA - Machine Learning cơ bản. Retrieved from <https://machinelearningcoban.com/tabml_book/ch_data_processing/eda.html>, 2024
6. Weizheng Yan, Han Zhang, Jing Sui, Dinggang Shen, Deep Chronnectome Learning via Full Bidirectional Long Short-Term Memory Networks for MCI Diagnosis. Retrieved from: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6553484/>