

ĐẠI HỌC BÁCH KHOA HÀ NỘI
KHOA TOÁN - TIN

ĐỒ ÁN I

PHÂN TÍCH SẮC THÁI BÌNH LUẬN SỬ DỤNG MÁY HỌC

ĐỒNG VĂN SỸ HOÀNG

Email: hoang.dvs227231@sis.hust.edu.vn

Mã số sinh viên: 20227231

Chuyên ngành Hệ thống Thông tin Quản lý

Giảng viên hướng dẫn: TS. Lê Hải Hà

Chữ kí GVHD

Hà Nội, 2025

NHẬN XÉT CỦA GIẢNG VIÊN

1. Mục tiêu và nội dung của đề án

- Mục tiêu: Xây dựng mô hình học máy và học sâu xử lý bài toán phân loại sắc thái bình luận.
- Nội dung:
 - Tìm hiểu nội dung về xử lý ngôn ngữ tự nhiên, lý thuyết về mô hình học máy và mô hình học sâu.
 - Xây dựng các mô hình cho bài toán phân loại sắc thái bình luận

2. Kết quả đạt được

- Đọc và nghiên cứu các báo cáo khoa học chuyên ngành, tài liệu tham khảo uy tín trên thế giới
- Xây dựng được mô hình SVM và PhoBERT để giải quyết bài toán

3. Ý thức làm việc của sinh viên:

- Sinh viên có tinh thần cầu tiến, có ý thức làm việc tốt, tham gia đầy đủ các buổi họp và đánh giá đề án

Hà Nội, ngày 10 tháng 6 năm 2025

Giảng viên hướng dẫn

TS. Lê Hải Hà

PHIẾU BÁO CÁO TIẾN ĐỘ

Danh sách đánh giá đồ án

Ngày đánh giá	Lần	Nội dung kế hoạch	Nội dung đã thực hiện	Điểm tích cực	Điểm nội dung	Ghi chú
11/04/2025	1	Theo kế hoạch	Tốt	10	10	
16/05/2025	2	Theo kế hoạch	Tốt	10	10	

LỜI CẢM ƠN

Trong suốt quá trình học tập và tìm hiểu nội dung đồ án, tác giả luôn nhận được sự quan tâm, hướng dẫn giúp đỡ tận tình của các thầy, cô giáo thuộc Khoa Toán - Tin cùng với sự góp ý từ bạn bè.

Đặc biệt, em xin gửi lời cảm ơn sâu sắc nhất đến TS. Lê Hải Hà. Trong suốt quá trình thực hiện đồ án I, sự hướng dẫn tận tình, những góp ý chuyên môn quý báu và sự động viên kịp thời của Thầy đã giúp em định hướng rõ ràng, vượt qua những khó khăn và hoàn thiện tốt nhất đồ án của mình. Những kiến thức và kinh nghiệm mà Thầy chia sẻ không chỉ giúp em trong phạm vi đồ án này mà còn là hành trang quý giá cho con đường học tập và nghiên cứu sau này. Em xin kính chúc Thầy luôn dồi dào sức khỏe, hạnh phúc và thành công trong sự nghiệp giảng dạy và nghiên cứu khoa học

Em xin chân thành cảm ơn!

Hà Nội, ngày 10 tháng 6 năm 2025

Sinh viên thực hiện đồ án

Đông Văn Sỹ Hoàng

TÓM TẮT ĐỒ ÁN

Đồ án này tập trung nghiên cứu và giải quyết bài toán phân loại cảm xúc cho văn bản tiếng Việt, một lĩnh vực có ý nghĩa thực tiễn cao trong việc phân tích ý kiến người dùng.

Nội dung đồ án được trình bày qua ba chương chính:

Chương 1: Giới thiệu tổng quan, đặt ra vấn đề, xác định mục tiêu là xây dựng và so sánh hiệu quả giữa hai phương pháp: học máy truyền thống và học sâu hiện đại. Chương cũng khoanh vùng đối tượng, phạm vi nghiên cứu và nêu bật ý nghĩa khoa học của đề tài.

Chương 2: Đi sâu vào cơ sở lý thuyết, cung cấp kiến thức nền tảng về các kỹ thuật được sử dụng. Chương trình bày chi tiết về phương pháp trích xuất đặc trưng TF-IDF, nguyên lý hoạt động của mô hình phân loại Support Vector Machine (SVM), và kiến trúc của mô hình ngôn ngữ tiên tiến PhoBERT dành riêng cho tiếng Việt.

Chương 3: Thực nghiệm trọng tâm, mô tả toàn bộ quá trình xây dựng và đánh giá mô hình. Bắt đầu từ việc thu thập, phân tích và tiền xử lý dữ liệu, chương trình bày song song quá trình huấn luyện hai mô hình: một mô hình kết hợp TF-IDF và SVM, và một mô hình tinh chỉnh từ PhoBERT. Cuối chương, các kết quả thu được từ hai mô hình được đánh giá và so sánh một cách chi tiết thông qua các độ đo hiệu năng, từ đó rút ra những nhận xét khách quan về ưu, nhược điểm của từng phương pháp.

Cuối cùng, đồ án đưa ra kết luận tổng kết lại các kết quả đã đạt được, khẳng định việc hoàn thành các mục tiêu đề ra và đề xuất một số Hướng phát triển tiềm năng để cải thiện và mở rộng đề tài trong tương lai.

Hà Nội, ngày 10 tháng 6 năm 2025

Sinh viên thực hiện đồ án

Đông Văn Sỹ Hoàng

Mục Lục

BẢNG THUẬT NGỮ TÊN VIẾT TẮT	3
DANH MỤC HÌNH ẢNH.....	4
DANH MỤC BẢNG.....	5
CHƯƠNG 1 GIỚI THIỆU CHUNG	6
1.1. Đặt vấn đề	6
1.2. Mục tiêu của đồ án	6
1.3. Đối tượng và phạm vi nghiên cứu.....	7
1.4. Phương pháp nghiên cứu	7
1.5. Ý nghĩa khoa học và thực tiễn	8
CHƯƠNG 2 TỔNG QUAN LÝ THUYẾT	9
2.1. Tổng quan về học máy	9
2.1.1. Embedding vector.....	9
2.1.2. Feature Engineering trong học máy	10
2.1.3. Phân loại cảm xúc bằng học máy có giám sát.....	10
2.2. TF-IDF	11
2.2.1. Term frequency (TF)	11
2.2.2. Inverse Document Frequency (IDF).....	12
2.2.3. TF-IDF	12
2.2.4. TF-IDF và liên kết với lý thuyết thông tin	13
2.3. Mô hình SVM – Support Vector Machine.....	14
2.3.1. Khoảng cách từ 1 điểm tới 1 siêu phẳng	14
2.3.2. Bài toán phân chia 2 lớp	15
2.3.3. Bài toán tối ưu cho SVM.....	16
2.3.4. Phân lớp lề mềm	18
2.4. Mô hình PhoBERT.....	20
2.4.1. Giới thiệu	20

2.4.2.	Kiến trúc mô hình PhoBERT.....	22
2.4.3.	Embedding trong PhoBERT.....	24
2.4.4.	Mask Language Model	25
CHƯƠNG 3 KIỂM TRA VÀ ĐÁNH GIÁ CÁC MÔ HÌNH.....		27
3.1.	Xử Lý Dữ Liệu Đầu Vào	27
3.1.1.	Dữ liệu sử dụng	27
3.1.2.	So sánh sự phân bố dữ liệu đầu vào	27
3.1.3.	Đọc và chuyển đổi dữ liệu.....	28
3.1.4.	Chuẩn hóa dữ liệu đầu vào	29
3.2.	Huấn Luyện Mô Hình TF-IDF Kết Hợp SVM	29
3.2.1.	Trích xuất vector cho model.....	29
3.2.2.	Lựa chọn tham số, chia validation.....	30
3.2.3.	Thử nghiệm model.....	30
3.3.	Huấn Luyện Mô Hình PhoBERT.....	30
3.3.1.	Xây dựng Tokenizer	30
3.3.2.	Huấn luyện mô hình với Masked Language Model	31
3.3.3.	Thử nghiệm model.....	31
3.4.	Đánh Giá Kết Quả Sau Khi Xây Dựng.....	31
3.4.1.	Mô hình SVM.....	31
3.4.2.	Mô hình PhoBERT	34
3.4.3.	So sánh giữa hai mô hình	35
KẾT LUẬN		38
TÀI LIỆU THAM KHẢO		39

BẢNG THUẬT NGỮ TÊN VIẾT TẮT

CRM	Customer Relationship Management
ID	Identifier
GLoVe	Global Vectors for Word Representation
Word2Vec	Word to Vector
BoW	Bag of Words
TF	Term Frequency
IDF	Inverse Document Frequency
SVM	Support Vector Machine
PLA	Perceptron Learning Algorithm
BERT	Bidirectional Representation for Transformers
GPT	Generative Pre-trained Transformer
NLU	Natural Language Understanding
BPE	Byte Pair Encoding
NER	Named Entity Recognition
XLM	Cross-lingual Language Model
XLM-R	XLM-RoBERTa
RoBERTa	Robustly Optimized BERT Approach
MLM	Masked Language Model
NSP	Next Sentence Prediction

DANH MỤC HÌNH ẢNH

Hình 2.1 Ví dụ về Feature Engineering	10
Hình 2.2 Cấu trúc của các mô hình học máy	10
Hình 2.3 Cách phân chia 2 classes linearly separable.....	15
Hình 2.4 Margin của 2 classes là bằng nhau và rộng nhất có thể	15
Hình 2.5 Phân tích bài toán SVM	17
Hình 2.6 Hạn chế của phân lớp lề cứng	19
Hình 2.7 So sánh phân lớp lề cứng và phân lớp lề mềm.....	19
Hình 2.8. So sánh Transformer-GPT-BERT	21
Hình 2.9 Kiến trúc của PhoBERT	23
Hình 2.10 Đầu vào của PhoBERT	24
Hình 3.1 Dữ liệu đầu vào	27
Hình 3.2 So sánh sự phân bố dữ liệu đầu vào	28
Hình 3.3 Chuẩn hóa dữ liệu đầu vào.....	29
Hình 3.4 Bộ tham số tối ưu của mô hình SVM.....	30
Hình 3.5 Ví dụ về sự phân loại của mô hình SVM.....	30
Hình 3.6 Xây dựng Tokenizer và huấn luyện PhoBERT	31
Hình 3.7 Ví dụ về sự phân loại của mô hình PhoBERT	31
Hình 3.8 Ma trận nhầm lẫn của mô hình SVM.....	32
Hình 3.9. Ma trận nhầm lẫn của mô hình PhoBERT	34

DANH MỤC BẢNG

Bảng 3.1 Hiệu suất đánh giá của mô hình SVM.....	33
Bảng 3.2 Hiệu suất đánh giá của mô hình PhoBERT	35
Bảng 3.3 So sánh hiệu suất của hai mô hình.....	36

PHẦN 1 GIỚI THIỆU CHUNG

1.1. Đặt vấn đề

Trong kỷ nguyên số hóa, sự bùng nổ của Internet và các nền tảng mạng xã hội như Facebook, TikTok, các sàn thương mại điện tử như Shopee, Tiki, Lazada đã tạo ra một khối lượng dữ liệu văn bản khổng lồ. Dữ liệu này, chủ yếu dưới dạng bình luận, đánh giá, và phản hồi, là nguồn thông tin vô giá, phản ánh một cách chân thực và tức thời những suy nghĩ, cảm xúc và quan điểm của người dùng về sản phẩm, dịch vụ hay các vấn đề xã hội.

Đối với các doanh nghiệp và tổ chức, việc khai thác hiệu quả nguồn dữ liệu này mang lại lợi thế cạnh tranh to lớn. Phân tích ý kiến khách hàng giúp doanh nghiệp cải thiện chất lượng sản phẩm, tối ưu hóa dịch vụ, theo dõi sức khỏe thương hiệu, phát hiện sớm các nguy cơ khủng hoảng truyền thông và nắm bắt xu hướng thị trường. Tuy nhiên, việc phân tích hàng triệu bình luận bằng phương pháp thủ công là một thách thức lớn, đòi hỏi chi phí nhân lực khổng lồ, tốn nhiều thời gian và kết quả dễ bị ảnh hưởng bởi yếu tố chủ quan của người phân tích.

Thêm vào đó, việc xử lý ngôn ngữ tự nhiên tiếng Việt chứa đựng nhiều thách thức đặc thù. Sự đa dạng trong cách diễn đạt, việc sử dụng từ lóng, teencode, các cấu trúc câu phức tạp, cùng với hiện tượng mỉa mai (sarcasm) và châm biếm (irony) khiến cho việc xác định chính xác sắc thái của một câu trở nên vô cùng khó khăn.

Từ những thực trạng trên, việc xây dựng một hệ thống tự động có khả năng "đọc hiểu" và phân loại sắc thái của các bình luận tiếng Việt một cách nhanh chóng và chính xác đã trở thành một nhu cầu cấp thiết. Bài toán này được biết đến trong lĩnh vực Khoa học Máy tính với tên gọi Phân tích sắc thái (Sentiment Analysis). Việc giải quyết bài toán này không chỉ giúp tự động hóa quy trình phân tích phản hồi mà còn cung cấp những insight khách quan và có giá trị, hỗ trợ đắc lực cho việc ra quyết định.

1.2. Mục tiêu của đề án

Đề án này tập trung vào việc xây dựng và đánh giá các mô hình học máy cho bài toán phân tích sắc thái bình luận tiếng Việt

- Mục tiêu chính: Xây dựng và so sánh hiệu quả của các mô hình có khả năng phân loại tự động bình luận tiếng Việt thành các nhãn sắc thái cụ thể
- Các mục tiêu cụ thể:
 - Nghiên cứu tổng quan về lĩnh vực xử lý ngôn ngữ tự nhiên, phân tích sắc thái và các kiến thức nền tảng về học máy, học sâu
 - Thu thập, xử lý và xây dựng bộ dữ liệu bình luận tiếng Việt phù hợp cho việc huấn luyện và đánh giá mô hình

- Triển khai và đánh giá hiệu năng của mô hình theo hướng tiếp cận học máy cổ điển: sử dụng phương pháp trích xuất đặc trưng TF-IDF (Term Frequency-Inverse Document Frequency) kết hợp với giải thuật phân loại SVM (Support Vector Machine).
- Triển khai và đánh giá hiệu năng của mô hình theo hướng tiếp cận học sâu hiện đại: sử dụng mô hình ngôn ngữ lớn đã được huấn luyện trước cho tiếng Việt là PhoBERT.
- So sánh, phân tích và đánh giá kết quả của hai hướng tiếp cận trên dựa vào các độ đo phổ biến như Accuracy, Precision, Recall và F1-Score để rút ra kết luận về ưu, nhược điểm của từng phương pháp.

1.3. Đối tượng và phạm vi nghiên cứu

- Đối tượng nghiên cứu: Các bình luận, đánh giá của người dùng bằng Tiếng Việt trên các nền tảng trực tuyến
- Phạm vi nghiên cứu:
 - **Về dữ liệu:** Đồ án tập trung vào việc phân tích dữ liệu văn bản được thu thập từ các bình luận công khai trên sàn thương mại điện tử Shopee
 - **Về bài toán:** Phân loại sắc thái ở cấp độ câu với 2 nhãn : **Tích cực** và **Tiêu cực**
 - **Về mô hình:** Đồ án không đi sâu vào việc tạo ra một mô hình kiến trúc mới mà tập trung vào việc ứng dụng, tinh chỉnh và so sánh 2 phương pháp TF-IDF kết hợp SVM và PhoBERT

1.4. Phương pháp nghiên cứu

Để đạt được các mục tiêu đề ra, đồ án sẽ sử dụng kết hợp các phương pháp nghiên cứu sau:

- Phương pháp nghiên cứu lý thuyết: Tổng hợp, phân tích và hệ thống hóa các tài liệu, bài báo khoa học, giáo trình liên quan đến xử lý ngôn ngữ tự nhiên, các giải thuật học máy (SVM), các mô hình học sâu (PhoBERT) để xây dựng cơ sở lý luận cho đồ án.
- Phương pháp nghiên cứu thực nghiệm: Tiến hành các thí nghiệm trên tập dữ liệu đã thu thập. Quá trình này bao gồm các bước: tiền xử lý dữ liệu, huấn luyện mô hình, tinh chỉnh tham số và đánh giá kết quả thông qua các độ đo định lượng.
- Phương pháp so sánh, phân tích: Dựa trên kết quả thực nghiệm, tiến hành so sánh hiệu năng giữa các mô hình đã triển khai để chỉ ra điểm mạnh, điểm yếu và khả năng ứng dụng của từng phương pháp trong bối cảnh bài toán cụ thể.

1.5. Ý nghĩa khoa học và thực tiễn

Đồ án này mang lại ý nghĩa quan trọng trên cả phương diện khoa học và ứng dụng thực tiễn. Về mặt khoa học, đề tài không chỉ góp phần hệ thống hóa các kiến thức chuyên sâu về bài toán phân tích sắc thái cho ngôn ngữ tiếng Việt, mà còn cung cấp một cái nhìn so sánh thực nghiệm, định lượng về hiệu quả giữa hướng tiếp cận học máy cổ điển (TF-IDF và SVM) và học sâu hiện đại (PhoBERT). Kết quả đánh giá khách quan này sẽ là một tài liệu tham khảo giá trị, đóng góp vào cơ sở tri thức chung cho các nghiên cứu liên quan trong tương lai. Về mặt thực tiễn, kết quả của đồ án là nền tảng vững chắc để xây dựng các ứng dụng tự động phân tích phản hồi khách hàng, giúp các doanh nghiệp tiết kiệm đáng kể chi phí, thời gian, nâng cao hiệu quả kinh doanh và đưa ra các quyết định chiến lược kịp thời. Mô hình được phát triển hoàn toàn có thể tích hợp liền mạch vào các hệ thống lớn hơn như Quản trị quan hệ khách hàng (CRM), chatbot thông minh hay các công cụ theo dõi sức khỏe thương hiệu, đồng thời trở thành một tài liệu hướng dẫn hữu ích cho cộng đồng sinh viên và các nhà phát triển đang quan tâm đến lĩnh vực xử lý ngôn ngữ tự nhiên tại Việt Nam.

PHẦN 2 TỔNG QUAN LÝ THUYẾT

2.1. Tổng quan về học máy

2.1.1. *Embedding vector*

Trong lĩnh vực Xử lý Ngôn ngữ Tự nhiên (NLP), một trong những thách thức cơ bản nhất là làm thế nào để máy tính, vốn chỉ làm việc với các con số, có thể "hiểu" và xử lý được văn bản. Con người hiểu từ ngữ thông qua ngữ nghĩa, mối liên hệ và ngữ cảnh, nhưng máy tính thì không. Embedding Vector (hay Word Embedding) ra đời như một giải pháp đột phá cho vấn đề này.

Về bản chất, Embedding Vector là một phương pháp biểu diễn từ hoặc câu dưới dạng một vector số thực, đa chiều và dày đặc (dense). Thay vì biểu diễn một từ bằng một ID đơn lẻ (ví dụ: "sinh viên" = 5) hay một vector One-Hot Encoding khổng lồ, chúng ta biểu diễn nó bằng một vector có số chiều xác định (ví dụ: 100, 300, hoặc 768 chiều).

Embedding không được thiết kế thủ công mà được "học" từ một kho dữ liệu văn bản lớn. Nguyên tắc cốt lõi đằng sau việc học này là **giả thuyết phân phối (distributional hypothesis)**: "những từ xuất hiện trong cùng một ngữ cảnh thì có xu hướng mang ngữ nghĩa giống nhau". Có hai họ phương pháp chính:

- **Embedding tĩnh (Static Embeddings)**: Các mô hình này gán một vector duy nhất cho mỗi từ trong bộ từ vựng, bất kể ngữ cảnh sử dụng của từ đó là gì. Thường dùng các mô hình:
 - **Word2Vec**: Là mô hình tiên phong, sử dụng mạng neural nông để học embedding
 - **GloVe(Global Vectors for Word Representation)**: Nó học embedding bằng cách tối ưu hóa dựa trên ma trận đồng xuất hiện của các từ trong kho dữ liệu
- **Embedding tĩnh (Contextual Embedding)**: Đây là bước tiến hóa vượt bậc, giải quyết điểm yếu của embedding tĩnh. Trong thực tế, một từ có thể có nhiều nghĩa tùy thuộc vào câu (ví dụ: từ "ngân hàng" trong "ngân hàng nhà nước" và "ngân hàng câu hỏi"). Embedding theo ngữ cảnh tạo ra một vector biểu diễn cho từ dựa trên chính câu mà nó xuất hiện.
 - **BERT** (Bidirectional Encoder Representations from Transformers): Là một trong những mô hình nổi bật nhất. BERT đọc toàn bộ câu cùng một lúc (bidirectional) để tạo ra vector biểu diễn cho mỗi từ, do đó vector này chứa đầy đủ thông tin ngữ cảnh của câu.
 - **PhoBERT**: Là một phiên bản BERT được huấn luyện trước (pre-trained) trên một kho dữ liệu tiếng Việt khổng lồ. Do đó, PhoBERT có

khả năng nắm bắt ngữ nghĩa và đặc thù của tiếng Việt một cách vượt trội, là lựa chọn hàng đầu cho các bài toán NLP tiếng Việt hiện nay.

2.1.2. Feature Engineering trong học máy

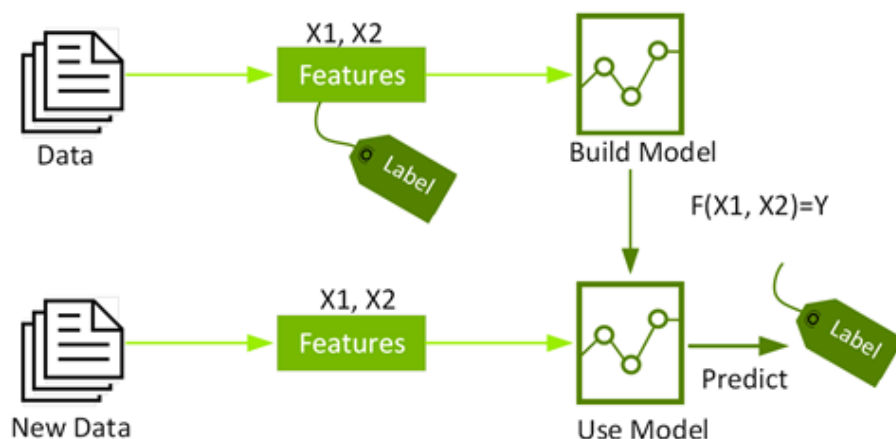
Feature Engineering là quá trình chuyển đổi dữ liệu thô thành đầu vào cho thuật toán học máy. Để được sử dụng trong các thuật toán học máy, các đặc điểm (feature) phải được đưa vào các vectors đặc trưng, là các vectors số đại diện cho giá trị của từng đặc điểm. Để phân tích cảm nghĩ, dữ liệu văn bản phải được đưa vào các vectors từ ngữ, là vectors của các số biểu thị giá trị cho mỗi ký tự. Văn bản đầu vào có thể được mã hóa thành các vectors ký tự bằng cách sử dụng các kỹ thuật đếm như Bag of Words (BoW), bag-of-ngrams hoặc Term Frequency/Inverse Document Frequency (TF-IDF).



Hình 2.1 Ví dụ về Feature Engineering

2.1.3. Phân loại cảm xúc bằng học máy có giám sát

Sau khi văn bản đầu vào đã được chuyển đổi thành vectors từ ngữ, thuật toán học máy phân loại có thể được sử dụng để phân loại cảm xúc. Phân loại là một nhóm các thuật toán học máy được giám sát để xác định chủng loại nào mà một item thuộc về (chẳng hạn như văn bản là tiêu cực hay tích cực) dựa trên dữ liệu được gắn nhãn (chẳng hạn như văn bản được gắn nhãn là tích cực hay tiêu cực).



Hình 2.2. Cấu trúc của các mô hình học máy

Các thuật toán học máy phân loại có thể được sử dụng để phân tích cảm nghĩ bao gồm một số loại :

- Naive Bayes là một bộ các thuật toán xác suất xác định có điều kiện của lớp dữ liệu đầu vào.
- Support Vector Machines tìm thấy một hyperlane trong không gian N chiều phân loại rõ ràng các điểm dữ liệu.
- Logistic regression sử dụng hàm logistic để mô hình hóa xác suất của một cấp nhất định.

2.2. TF-IDF

Trong khai thác thông tin, *tf-idf* (hoặc $TF \cdot IDF$, $TFIDF$, $TF-IDF$, hoặc $Tf-idf$), viết tắt của tần suất từ khóa - tần suất ngược tài liệu, là một thước đo mức độ quan trọng của một từ đối với một tài liệu trong một bộ sưu tập hoặc tập hợp tài liệu, được điều chỉnh để phản ánh thực tế rằng một số từ xuất hiện thường xuyên hơn trong các tài liệu nói chung. Giống như mô hình *bag-of-words*¹, nó mô hình hóa một tài liệu như một tập hợp các từ, không quan tâm đến thứ tự của từ. Nó là một sự cải tiến so với mô hình *bag-of-word* đơn giản, cho phép trọng số của các từ phụ thuộc vào phần còn lại của tập hợp tài liệu.

Nó thường được sử dụng như một yếu tố trọng số trong các tìm kiếm khai thác thông tin, khai thác văn bản và mô hình người dùng. Một cuộc khảo sát được thực hiện vào năm 2015 cho thấy 83% hệ thống đề xuất dựa trên văn bản trong các thư viện số sử dụng *tf-idf*. Các biến thể của sơ đồ trọng số *tf-idf* thường được các công cụ tìm kiếm sử dụng như một công cụ chính trong việc tính toán và xếp hạng mức độ liên quan của một tài liệu đối với truy vấn của người dùng.

Một trong những hàm xếp hạng đơn giản nhất được tính bằng cách cộng *tf-idf* cho mỗi từ khóa trong truy vấn; nhiều hàm xếp hạng tinh vi hơn là các biến thể của mô hình đơn giản này. $TF-IDF$ là tích của hai thống kê: tần suất từ (*term frequency*) và tần suất nghịch đảo của tài liệu (*inverse document frequency*). Có nhiều cách khác nhau để xác định giá trị chính xác của cả hai thống kê này.

2.2.1. Term frequency (TF)

Term frequency là tần số xuất hiện của một từ trong đoạn văn bản

$$tf(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \quad PT\ 2.1$$

¹https://www.researchgate.net/publication/338511771_An_Overview_of_Bag_of_WordsImportance_Implementation_Applications_and_Challenges

Trong đó :

- $f_{t',d}$: số lần xuất hiện của từ t trong văn bản d
- $\sum_{t' \in d} f_{t',d}$: tổng số lần xuất hiện của các từ bất kì trong văn bản

Hạn chế của TF :

- TF không tính đến mức độ quan trọng trong toàn cục của một thuật ngữ trong toàn bộ tập dữ liệu
- Những từ phổ biến như “là” hoặc “và” có thể có điểm TF cao nhưng không có ý nghĩa trong việc phân biệt các tài liệu

2.2.2. Inverse Document Frequency (IDF)

IDF giảm trọng số của các từ phổ biến trong nhiều tài liệu , đồng thời tăng trọng số của các từ hiếm. Nếu một thuật ngữ xuất hiện trong ít tài liệu hơn, nó có nhiều khả năng mang ý nghĩa và có tính đặc thù cao hơn.

$$idf(t, D) = \log \frac{N}{|\{d: d \in D \text{ và } t \in d\}|} \quad \text{PT 2.2}$$

Trong đó :

- N là tổng số văn bản trong tập D
- $|\{d: d \in D \text{ và } t \in d\}|$ là số văn bản mà từ t xuất hiện. Nếu từ t không xuất hiện ở bất kỳ văn bản nào trong tập thì mẫu số $= 0$, thì phép chia sẽ lỗi, vì thế người ta thường thay thế bằng $1 + |\{d: d \in D \text{ và } t \in d\}|$

Hạn chế của IDF

- IDF không xét đến tần suất xuất hiện của một thuật ngữ trong một tài liệu cụ thể
- Một thuật ngữ có thể hiếm trong toàn bộ tập dữ liệu (IDF cao) nhưng không quan trọng trong một tài liệu cụ thể

2.2.3. TF-IDF

$$tfidf(t, d, D) = tf(t, d) * idf(t, D) \quad \text{PT 2.3}$$

Trọng số cao trong tf-idf đạt được khi một thuật ngữ có tần suất xuất hiện cao (trong tài liệu cụ thể) và tần suất tài liệu thấp (trong toàn bộ tập hợp tài liệu); do đó, các trọng số có xu hướng loại bỏ các thuật ngữ phổ biến.

Vì tỷ số bên trong hàm logarit của IDF luôn lớn hơn hoặc bằng 1, giá trị của IDF và TI-IDF luôn lớn hơn hoặc bằng 0. Khi một thuật ngữ xuất hiện trong nhiều tài liệu hơn, tỷ số bên trong hàm logarit tiến dần đến 1, làm cho IDF và TF-IDF tiến gần hơn đến 0.

2.2.4. TF-IDF và liên kết với lý thuyết thông tin

Cả tần suất từ (TF) và tần suất nghịch đảo tài liệu (IDF) đều có thể được xây dựng dưới dạng lý thuyết thông tin; điều này giúp hiểu tại sao tích của chúng lại có ý nghĩa về nội dung thông tin chung của một tài liệu. Một giả định đặc trưng về phân phối $p(d,t)$ là:

$$p(d|t) = \frac{1}{|\{d \in D: t \in d\}|} \quad \text{PT 2.4}$$

Entropy có điều kiện của một tài liệu được chọn ngẫu nhiên trong tập hợp tài liệu D , với điều kiện tài liệu đó chứa một thuật ngữ cụ thể t (và giả định rằng tất cả các tài liệu đều có xác suất được chọn như nhau), được tính như sau:

$$\begin{aligned} H(D|T = t) &= - \sum_d p_{d|t} \log p_{d|t} & \text{PT 2.5} \\ &= - \log \frac{1}{|\{d \in D: t \in d\}|} \\ &= \log \frac{|\{d \in D: t \in d\}|}{|D|} \\ &+ \log |D| = -idf(t) + \log |D| \end{aligned}$$

Về kí hiệu, D và T là các biến ngẫu nhiên tương ứng với việc chọn một tài liệu hoặc một thuật ngữ. Thông tin tương hỗ có thể được biểu diễn dưới dạng:

$$\begin{aligned} M(T; D) &= H(D) - H(D|T) & \text{PT 2.6} \\ &= \sum_t p_t \cdot (H(D) - H(D|W = t)) \\ &= \sum_t p_t \cdot idf(f) \end{aligned}$$

Bước cuối cùng là mở rộng p_t , xác suất vô điều kiện để chọn một thuật ngữ, dựa trên việc chọn một tài liệu ngẫu nhiên, để thu được:

$$\begin{aligned}
M(T|D) &= \sum_{t,d} p_{t,d} \cdot p_t \cdot idf(t) & PT 2.7 \\
&= \sum_{t,d} tf(t,d) \cdot \frac{1}{|D|} \cdot idf(t) \\
&= \frac{1}{|D|} \sum_{t,d} tf(t,d) \cdot idf(t)
\end{aligned}$$

Biểu thức này cho thấy rằng tổng của tất cả giá trị tf-idf của mọi thuật ngữ và tài liệu khôi phục thông tin tương hỗ giữa các tài liệu và thuật ngữ, đồng thời tính đến tất cả các đặc điểm cụ thể trong phân phối chung của chúng.

Mỗi giá trị tf-idf do đó mang theo “bit thông tin” gắn với một cặp thuật ngữ, tài liệu

2.3. Mô hình SVM – Support Vector Machine

2.3.1. Khoảng cách từ 1 điểm tới 1 siêu phẳng

Trong không gian 2 chiều, ta biết được khoảng cách từ một điểm có tọa độ (x_0, y_0) tới đường thẳng có phương trình $w_1x + w_2y + b = 0$ được xác định bởi :

$$\frac{|w_1x_0 + w_2y_0 + b|}{\sqrt{w_1^2 + w_2^2}} \quad PT 2.8$$

Trong không gian 3 chiều, ta biết được khoảng cách từ một điểm có tọa độ (x_0, y_0, z_0) tới một mặt phẳng có phương trình $w_1x + w_2y + w_3z + b = 0$ được xác định bởi:

$$\frac{|w_1x_0 + w_2y_0 + w_3z_0 + b|}{\sqrt{w_1^2 + w_2^2 + w_3^2}} \quad PT 2.9$$

Trong trường hợp tổng quát, có thể lên không gian nhiều chiều: Khoảng cách từ một điểm (vector) có tọa độ x_0 tới siêu mặt phẳng (hyperplane) có phương trình $w^T x_0 + b = 0$ được xác định bởi:

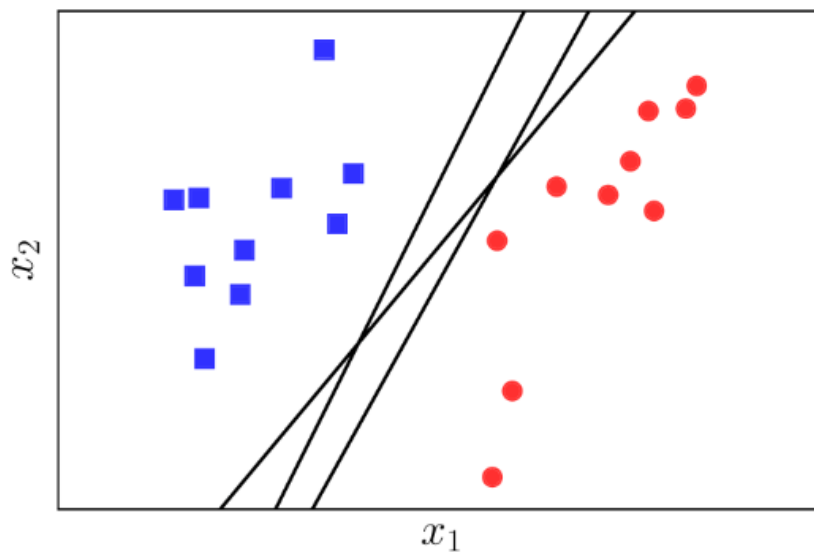
$$\frac{w^T x_0 + b}{\|w\|_2} \quad PT 2.10$$

Với $\|w\|_2 = \sqrt{\sum_{i=1}^d w_i^2}$ với d là số chiều của không gian.

2.3.2. Bài toán phân chia 2 lớp

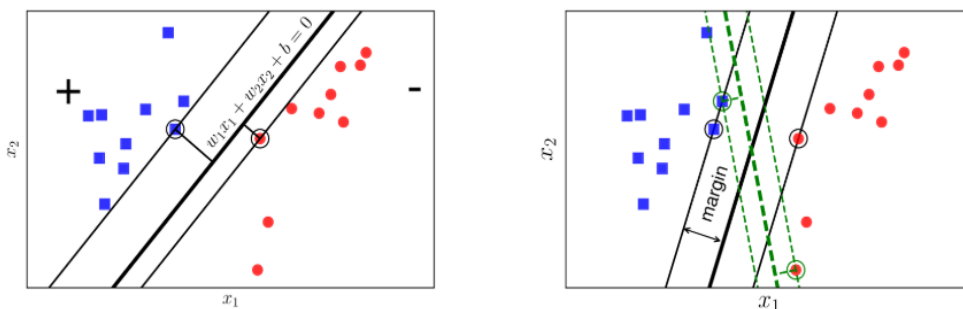
Giả sử rằng có hai class khác nhau được mô tả bởi các điểm trong không gian nhiều chiều, hai classes này linearly separable, tức tồn tại một siêu phẳng phân chia chính xác hai classes đó. Hãy tìm một siêu mặt phẳng phân chia hai classes đó, tức tất cả các điểm thuộc một class nằm về cùng một phía của siêu mặt phẳng đó và ngược phía với toàn bộ các điểm thuộc class còn lại.

Chúng ta đã biết rằng, *thuật toán PLA*² có thể làm được việc này nhưng nó có thể cho chúng ta vô số nghiệm như Hình 1 dưới đây:



Hình 2.3 Cách phân chia 2 classes linearly separable

Câu hỏi đặt ra là: Liệu có một mặt phẳng là tốt nhất nếu theo một tiêu chuẩn nào đó



Hình 2.4 Margin của 2 classes là bằng nhau và rộng nhất có thể

Ở hình trên, dễ thấy lớp tròn đỏ có khoảng cách đến đường phân chia nhỏ hơn so với lớp vuông xanh. Điều này không đảm bảo sự công bằng. Ta cần một đường phân chia

² <https://machinelearningcoban.com/2017/01/21/perceptron/>

sao cho khoảng cách từ điểm gần nhất của mỗi lớp tới đường phân chia là như nhau. Khoảng cách như nhau này được gọi là lề (*margin*).

Chúng ta xét tiếp Hình 2 bên phải khi khoảng cách từ đường phân chia tới các điểm gần nhất của mỗi class là như nhau. Xét hai cách phân chia bởi đường nét liền màu đen và đường nét đứt màu lục, đường nào sẽ làm cho cả hai class công bằng hơn? Rõ ràng đó phải là đường nét liền màu đen vì nó tạo ra một *margin* rộng hơn.

Việc *margin* rộng hơn sẽ mang lại hiệu ứng phân lớp tốt hơn vì *sự phân chia giữa hai classes là rạch ròi hơn*. Việc này, sau này các bạn sẽ thấy, là một điểm khá quan trọng giúp *Support Vector Machine* mang lại kết quả phân loại tốt hơn so với *Neural Network*³ với 1 layer, tức Perceptron Learning Algorithm.

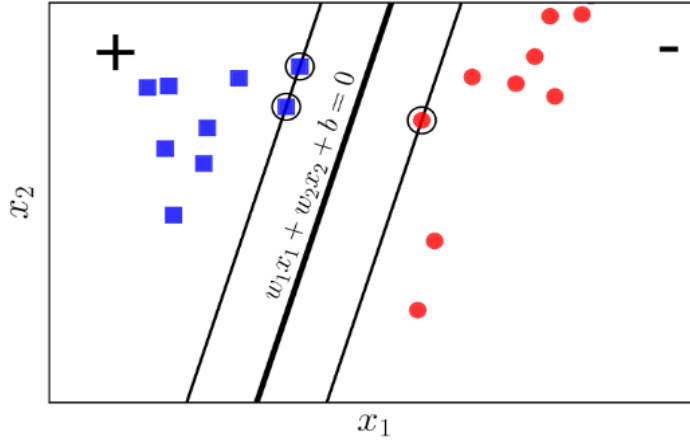
Bài toán tối ưu trong *Support Vector Machine* (SVM) chính là bài toán đi tìm đường phân chia sao cho *margin* là lớn nhất. Đây cũng là lý do vì sao SVM còn được gọi là *Maximum Margin Classifier*. Nguồn gốc của tên gọi Support Vector Machine sẽ sớm được làm sáng tỏ.

2.3.3. Bài toán tối ưu cho SVM

Giả sử rằng các cặp dữ liệu của training set là $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ với vector $x_i \in R^d$ thể hiện đầu vào của một điểm dữ liệu và y_i nhãn của điểm dữ liệu đó. d là số chiều của dữ liệu và N là số điểm dữ liệu. Giả sử rằng nhãn của mỗi điểm dữ liệu được xác định bởi $y_1 = 1$ (class 1) hoặc $y_2 = -1$ (class 2) giống như trong PLA

Để giúp các bạn dễ hình dung, chúng ta cùng xét trường hợp trong không gian hai chiều dưới đây. Không gian hai chiều để dễ hình dung, các phép toán hoàn toàn có thể được tổng quát lên không gian nhiều chiều.

³ <https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/>



Hình 2.5 Phân tích bài toán SVM

Giả sử rằng các điểm vuông xanh thuộc class 1, các điểm tròn đỏ thuộc class -1 và mặt $w^T x + b = w_1 x_1 + w_2 x_2 + b = 0$ là mặt phân cách hai classes. Hơn nữa, class 1 nằm về phía dương, class -1 nằm về phía âm của mặt phân chia. Nếu ngược lại, ta chỉ cần đổi dấu của \mathbf{w} và \mathbf{b} . Chú ý rằng chúng ta cần đi tìm các hệ số \mathbf{w} và \mathbf{b} .

Ta quan sát thấy một điểm quan trọng sau đây: với cặp dữ liệu (x_n, y_n) bất kỳ, khoảng cách từ điểm đó tới mặt phân chia là:

$$\frac{y_n(w^T x_n + b)}{\|w\|_2} \quad \text{PT 2.11}$$

Điều này có thể dễ nhận thấy vì theo giả sử ở trên, y_n luôn cùng dấu với phía của x_n . Từ đó suy ra y_n cùng dấu với $(w^T x_n + b)$, và tử số luôn là 1 số không âm

Với mặt phân chia như trên, margin được tính là khoảng cách gần nhất từ 1 điểm tới mặt đó (bất kể điểm nào trong hai classes)

$$\text{margin} = \min_n \frac{y_n(w^T x_n + b)}{\|w\|_2} \quad \text{PT 2.12}$$

Tập hợp các điểm nằm gần nhất với một đường biên sẽ giúp xác định phương trình đường biên nên chúng còn được gọi là tập hợp các điểm hỗ trợ (support points), ký hiệu là S . Trong hình 3, các điểm được khoanh tròn là các điểm thuộc tập hỗ trợ. Để tìm ra đường biên có độ rộng lề là lớn nhất, chúng ta cần tối đa hóa khoảng cách từ các điểm thuộc tập hỗ trợ tới đường biên. Điều này tương đương với giải bài toán tối ưu:

$$(w, b) = \arg \max_{w, b} \left\{ \min_n \frac{y_n(w^T x_n + b)}{\|w\|_2} \right\} = \arg \max_{w, b} \left\{ \frac{1}{\|w\|_2} \min_n y_n(w^T x_n + b) \right\} \quad \text{PT 2.13}$$

Nhận xét quan trọng nhất là nếu ta thay vector hệ số \mathbf{w} bởi $\mathbf{k}\mathbf{w}$ và \mathbf{b} bởi $\mathbf{k}\mathbf{b}$ trong đó \mathbf{k} là một hằng số dương thì mặt phân chia không thay đổi, tức là khoảng cách từ từng điểm đến mặt phân chia không đổi, tức margin không đổi. Dựa trên tính chất này, ta có thể giả sử:

$$y_n(w^T x_n + b) = 1 \quad \text{PT 2.14}$$

Như vậy với mọi n , ta có:

$$y_n(w^T x_n + b) \geq 1 \quad \text{PT 2.15}$$

Vậy bài toán tối ưu PT 2.13 có thể đưa về bài toán tối ưu có ràng buộc sau đây:

$$(w, b) = \arg \min_{w, b} \frac{1}{2} \|w\|_2^2 \quad \text{PT 2.16}$$

Ràng buộc: $y_n(w^T x_n + b) \geq 1, \forall n = 1, 2, \dots, N$

Bằng 1 biến đổi đơn giản, nghịch đảo hàm mục tiêu ta có thể đưa bài toán này về bài toán dưới đây:

$$(w, b) = \arg \min_{w, b} \frac{1}{2} \|w\|_2^2 \quad \text{PT 2.17}$$

Ràng buộc: $1 - y_n(w^T x_n + b) \leq 0, \forall n = 1, 2, \dots, N$

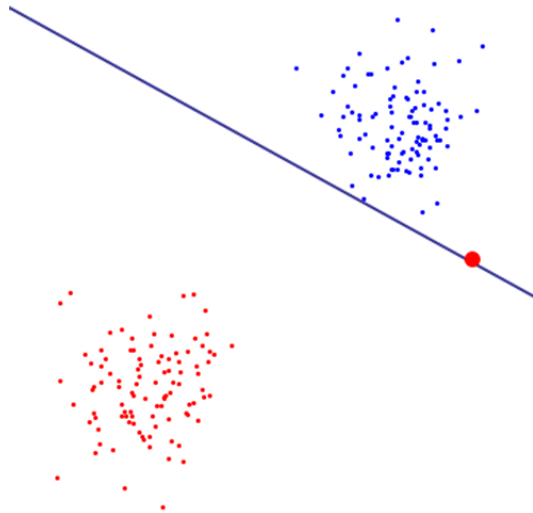
Quan sát quan trọng: Trong bài toán trên, hàm mục tiêu là một hàm norm, nên là một hàm lồi. Các hàm bất đẳng thức ràng buộc là các hàm tuyến tính theo w và b nên chúng cũng là một hàm lồi. Vậy bài toán tối ưu (3) có hàm mục tiêu là hàm lồi, các hàm ràng buộc cũng là lồi, nên nó là một bài toán lồi. Thậm chí, hàm mục tiêu là strictly convex vì

$\|w\|_2^2 = w^T \mathbf{I} w$ và \mathbf{I} là ma trận đơn vị - là ma trận xác định dương. Từ đây có thể suy ra nghiệm cho SVM là duy nhất.

2.3.4. Phân lớp lề mềm

Đường biên của thuật toán SVM sẽ chịu ảnh hưởng bởi các điểm thuộc tập hỗ trợ S . Trường hợp đường biên phân chia đúng mọi điểm dữ liệu được gọi là bài toán phân lớp lề cứng (*Hard Margin Classification*⁴). Tuy nhiên, phân lớp lề cứng gặp phải hạn chế khi xuất hiện dữ liệu ngoại lai

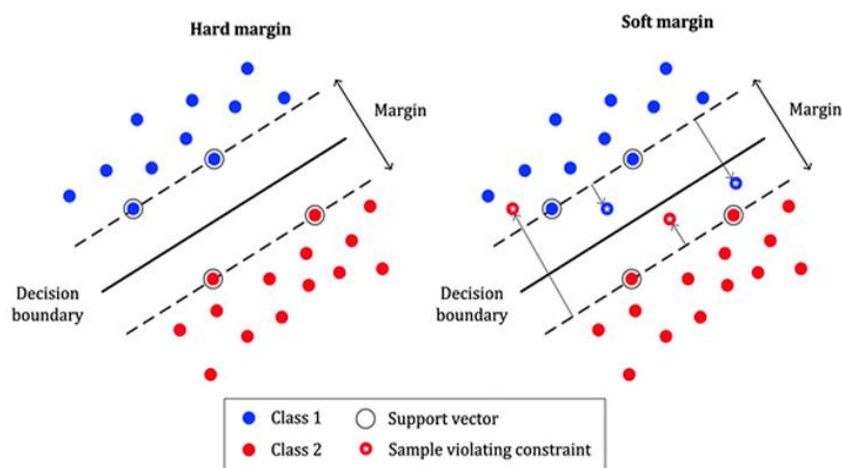
⁴ <https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/>



Hình 2.6. Hạn chế của phân lớp lề cứng

Phương pháp phân lớp lề cứng phân loại đúng mọi điểm dữ liệu, gồm cả dữ liệu ngoại lai. Điều này làm cho đường biên phân chia trở nên hẹp hơn. Khi đó, quy luật phân chia mất đi tính tổng quát và dẫn đến hiện tượng quá khớp (*overfitting*⁵). Kết quả dự đoán trên tập kiểm tra sẽ kém hơn so với tập huấn luyện.

Để khắc phục hạn chế của phân lớp lề cứng, kỹ thuật phân lớp lề mềm (*Soft Margin Classification*⁶) chấp nhận đánh đổi và cho phép phân lớp sai các điểm ngoại lai. Ta sẽ điều chỉnh cách tiếp cận để cho phép các điểm dữ liệu nằm ở sai phía của đường biên, nhưng với "hình phạt" tăng dần theo khoảng cách từ đường biên đó.



Hình 2.7 So sánh phân lớp lề cứng và phân lớp lề mềm

⁵ <https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/>

⁶ <https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/>

Ý tưởng của phân lớp lề mềm là mở rộng lề và chấp nhận một số điểm dữ liệu rơi vào vùng không an toàn để tạo ra một đường biên phân chia tổng quát hơn. Quá trình mở rộng lề sẽ bị giới hạn sao cho đối với những điểm rơi vào vùng không an toàn, tổng khoảng cách của chúng tới mép lề về phía mặt phẳng của nhãn đúng. Khoảng cách này được thể hiện qua biến slack (ký hiệu ξ).

Với $\xi_n \geq 0, n = 1, \dots, n$, mỗi biến slack ứng với một điểm dữ liệu huấn luyện. $\xi_n = 0$ khi các điểm dữ liệu nằm phía trên hoặc phía bên đúng của đường biên và $\xi_n = |W^T X_n + b - y_n|$ đối với các điểm còn lại. Do đó, một điểm dữ liệu nằm phía trên đường biên $W^T X_n + b = 0$ sẽ có $\xi_n = 1$, các điểm dữ liệu với $\xi_n \geq 1$ sẽ bị phân lớp sai. Ràng buộc 3 sẽ được thay thế bằng :

$$y_n(W^T X_n + b) \geq 1 - \xi_n, \forall n = 1, \dots, n \quad \text{PT 2.18}$$

Trong đó các biến slack bị giới hạn để thỏa mãn $\xi_n \geq 0$. Các điểm dữ liệu với $\xi_n = 0$ được phân lớp đúng và nằm trên hoặc phía bên đúng của lề. Các điểm dữ liệu với $0 < \xi_n < 1$ nằm phía trong lề, nhưng phía bên đúng của đường biên phân chia. Các điểm với $\xi_n > 1$ nằm phía bên sai của đường biên phân chia và được coi là phân lớp sai.

Mục tiêu là tối đa hóa độ rộng của lề đồng thời áp dụng hình phạt cho các điểm nằm ở phía sai của lề. Ta sẽ tối thiểu hóa:

$$C \sum_{n=1}^N \xi_n + \frac{1}{2} \|w\|_2^2 \quad \text{PT 2.19}$$

Từ đó ta có hàm loss như sau:

$$\ell(w, b) = \frac{1}{2} \|w\|_2^2 + C \sum_{i=1}^N \max(0, 1 - (w^T x^i + b)t^i) \quad \text{PT 2.20}$$

Trong đó tham số $C > 0$ có kiểm soát sự đánh đổi giữa biến slack và lề. Vì bất kỳ điểm nào bị phân lớp sai đều có $\xi_n > 1$ do đó $\sum_{n=1}^N \xi_n$ sẽ đóng vai trò là một giới hạn trên của số điểm bị phân loại sai.

2.4. Mô hình PhoBERT

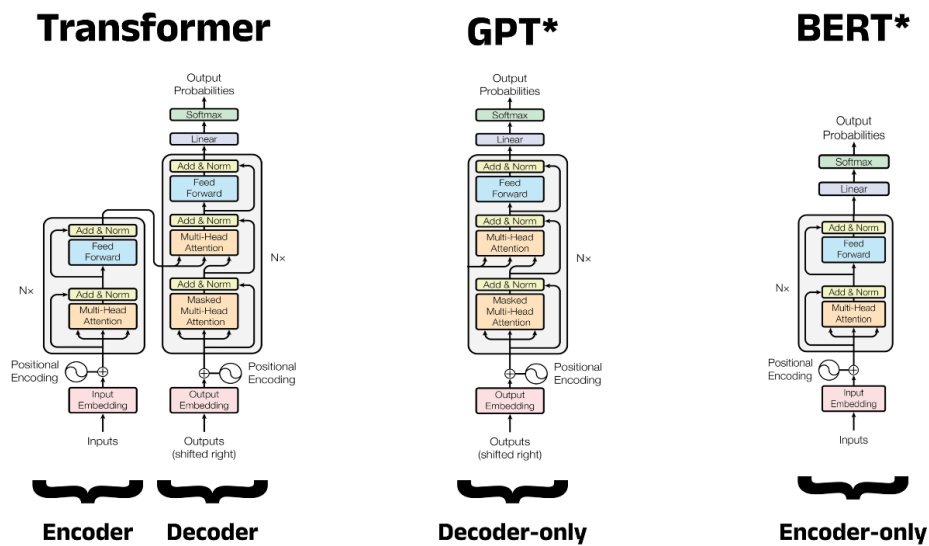
2.4.1. Giới thiệu

BERT là viết tắt của *Bidirectional Representation for Transformers* (Biểu diễn hai chiều cho Transformers) và được đề xuất bởi các nhà nghiên cứu tại Google AI Language vào năm 2018. Mặc dù mục tiêu chính ban đầu là nhằm cải thiện khả năng hiểu ý nghĩa của các truy vấn liên quan đến Tìm kiếm Google, BERT đã nhanh chóng trở thành một trong những kiến trúc quan trọng và toàn diện nhất cho nhiều tác vụ xử lý

ngôn ngữ tự nhiên (NLP), đồng thời đạt được kết quả vượt trội (*state-of-the-art*) trong các tác vụ như phân loại cặp câu, trả lời câu hỏi, v.v.

BERT là một kỹ thuật mạnh mẽ trong xử lý ngôn ngữ tự nhiên, có thể giúp máy tính hiểu ngôn ngữ con người tốt hơn. Nền tảng của BERT là khai thác ngữ cảnh hai chiều để học các biểu diễn từ và cụm từ phức tạp và sâu sắc. Bằng cách đồng thời xem xét cả ngữ cảnh phía trước và phía sau của một từ, BERT có thể nắm bắt toàn bộ ý nghĩa của từ đó trong ngữ cảnh, khác với các mô hình trước đây chỉ xét đến ngữ cảnh bên trái hoặc bên phải.

Điều này giúp BERT xử lý được các hiện tượng ngôn ngữ phức tạp và mơ hồ như đa nghĩa (polysemy), đồng tham chiếu (co-reference), và các mối quan hệ ngữ nghĩa tầm xa.



*Illustrative example, exact model architecture may vary slightly

Hình 2.8. So sánh Transformer-GPT-BERT

PhoBERT là một mô hình xử lý ngôn ngữ tự nhiên (NLP) dành riêng cho tiếng Việt, được phát triển dựa trên kiến trúc **BERT (Bidirectional Encoder Representations from Transformers)** của Google. Đây là mô hình đầu tiên áp dụng kiến trúc BERT được huấn luyện tiền đào tạo (pre-trained) trên tập dữ liệu lớn gồm văn bản tiếng Việt. PhoBERT được phát triển bởi nhóm nghiên cứu tại **VietAI**, với mục tiêu cải thiện hiệu suất của các tác vụ NLP trên tiếng Việt.

Trước khi PhoBERT ra đời, hầu hết các mô hình ngôn ngữ đơn ngữ cho tiếng Việt đều được huấn luyện trên corpus Wikipedia tiếng Việt ⁷— đây là nguồn dữ liệu duy nhất được sử dụng trong nghiên cứu của Vu et al. (2019), đồng thời cũng là nguồn tiếng Việt duy nhất được tích hợp vào dữ liệu tiền huấn luyện của các mô hình đa ngữ như mBERT hay XLM, ngoại trừ XLM-R. Tuy nhiên, dữ liệu từ Wikipedia không phản ánh đầy đủ cách sử dụng ngôn ngữ trong đời sống hàng ngày, đồng thời dung lượng dữ liệu cũng rất hạn chế (chỉ khoảng 1GB sau khi giải nén). Trong khi đó, các nghiên cứu trước (Liu et al., 2019) đã cho thấy rằng việc mở rộng quy mô dữ liệu tiền huấn luyện có thể giúp cải thiện đáng kể hiệu quả của mô hình ngôn ngữ.

Một vấn đề lớn khác trong xử lý tiếng Việt là đặc thù cấu trúc ngôn ngữ, cụ thể là sự mơ hồ giữa âm tiết và từ. Trong tiếng Việt, khoảng trắng không chỉ phân tách các từ mà còn dùng để phân tách các âm tiết, dẫn đến việc nhiều mô hình không thể nhận diện đúng ranh giới từ. Ví dụ, câu “Tôi là một nghiên cứu viên” gồm 6 âm tiết nhưng chỉ tạo thành 4 từ: “Tôi”, “là”, “một”, “nghiên_cứu_viên”.

Tuy nhiên, các mô hình ngôn ngữ BERT hiện có (cả đơn ngữ lẫn đa ngữ) đều không thực hiện bước tách từ tiếng Việt trước khi áp dụng phương pháp mã hóa BPE (Byte-Pair Encoding), mà trực tiếp áp dụng BPE lên chuỗi âm tiết, điều này có thể làm giảm hiệu quả khi áp dụng vào các tác vụ xử lý ngôn ngữ ở mức từ.

2.4.2. Kiến trúc mô hình PhoBERT

Về cốt lõi, PhoBERT không phát triển một kiến trúc hoàn toàn mới mà kế thừa và cải tiến từ RoBERTa, một phiên bản tối ưu của BERT. Do đó, kiến trúc nền tảng của PhoBERT chính là kiến trúc của RoBERTa.

Khối encoder trong kiến trúc Transformer có nhiệm vụ nhận vào một chuỗi đầu vào và tạo ra các biểu diễn vector số giàu thông tin cho mỗi từ (chính xác hơn là mỗi token). Mô hình chỉ gồm encoder như PhoBERT sẽ loại bỏ phần decoder và xếp chồng nhiều encoder lại với nhau để tạo thành một mô hình hoàn chỉnh. Những mô hình này không xử lý "prompt" để sinh văn bản một cách tự do, mà nhận một chuỗi đầu vào để đưa ra dự đoán (ví dụ: dự đoán một từ bị thiếu trong chuỗi). Do không có decoder – thành phần dùng để tạo ra từ mới – nên các mô hình này không được sử dụng cho các ứng dụng chatbot sinh văn bản dài như GPT. Thay vào đó, mô hình chỉ gồm encoder như PhoBERT rất mạnh cho các nhiệm vụ hiểu ngôn ngữ tự nhiên (NLU) như: nhận dạng thực thể có tên (NER), phân loại văn bản, và phân tích cảm xúc, đặc biệt là với tiếng Việt.

⁷ Vu, Xuan-Son, et al. (2019). [ETNLP: A visual-aided systematic approach to select pre-trained embeddings for a downstream task...](#)

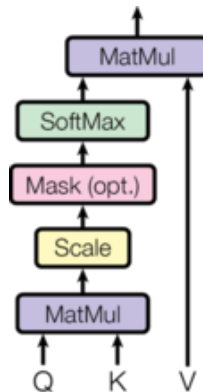
Các biểu diễn vector giàu thông tin được tạo ra bởi các khối encoder chính là thứ giúp PhoBERT có khả năng hiểu sâu sắc văn bản tiếng Việt. Kiến trúc của PhoBERT là một bộ mã hóa Transformer nhiều lớp theo hai chiều (multi-layer bidirectional Transformer encoder), tương tự như BERT. Hai cấu hình mô hình chủ yếu được VinAI Research công bố là:

- $PhoBERT_{BASE}$: $L = 12, H = 768, A = 12$, Tổng tham số = 135 triệu
- $PhoBERT_{LARGE}$: $L = 24, H = 1024, A = 16$, Tổng tham số = 370 triệu

Trong đó

- L : số lớp (layers)
- H: kích thước vector ẩn (hidden size)
- A: số đầu self-attention

Cơ chế Attention: Giống hệt BERT, PhoBERT sử dụng self-attention⁸ hai chiều (bidirectional), cho phép mỗi token trong câu có thể chú ý đến tất cả các token khác (cả bên trái và bên phải nó). Điều này giúp mô hình xây dựng được một biểu diễn ngữ cảnh toàn diện cho mỗi từ, một yếu tố cực kỳ quan trọng để hiểu đúng ngữ nghĩa của câu.



Hình 2.9 Kiến trúc của PhoBERT

Trong thực tế, chúng ta tính hàm attention trên một tập các truy vấn (queries) cùng lúc, được gom lại thành một ma trận Q. Các khóa (keys) và giá trị (values) cũng được gom lại thành các ma trận K và V. Chúng ta tính ma trận đầu ra theo công thức :

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad PT\ 2.21$$

Multi-head attention cho phép mô hình đồng thời chú ý đến thông tin từ các không gian biểu diễn khác nhau tại các vị trí khác nhau. Với chỉ một “head” attention duy nhất, việc lấy trung bình sẽ làm mất đi khả năng này

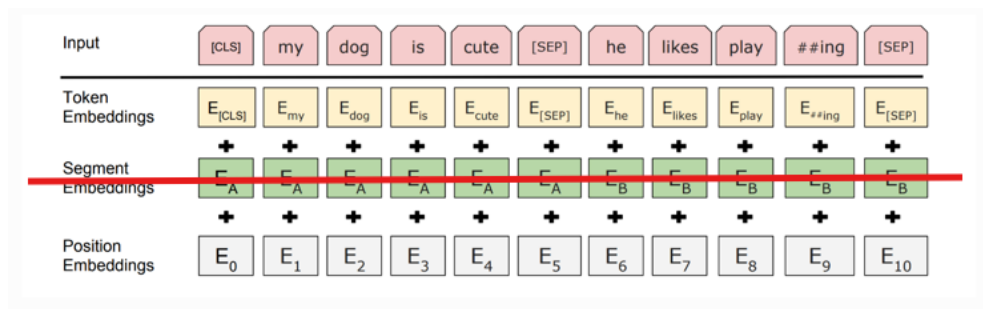
$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_h)W^O$$

Trong đó $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$

2.4.3. Embedding trong PhoBERT

Sự khác biệt quan trọng và tinh tế nhất của PhoBERT so với BERT nằm ở cách nó biểu diễn đầu vào thông qua lớp nhúng. Để xử lý một chuỗi văn bản, PhoBERT trước tiên chuyển nó thành một chuỗi token và sau đó tạo ra một vector biểu diễn đầu vào cho mỗi token bằng cách tổng hợp các thành phần nhúng.

Để mô hình có thể xử lý văn bản, chuỗi ký tự đầu vào phải được chuyển đổi thành một chuỗi các vector số. Quá trình này trong PhoBERT được thực hiện qua nhiều bước và có những đặc điểm riêng biệt.



Hình 2.10 Đầu vào của PhoBERT

- Tiền xử lý và Tokenization
 - **Tách từ** : Một yêu cầu đặc thù của PhoBERT là văn bản đầu vào cần phải được tách từ trước khi đưa vào tokenizer. Ví dụ, cụm từ "sinh viên" phải được chuẩn hóa thành "sinh_viên".
 - **Byte-Pair Encoding (BPE)**: Sau khi tách từ, chuỗi văn bản được mã hóa bằng tokenizer BPE⁹. PhoBERT sử dụng một bộ từ vựng (vocabulary) gồm khoảng 64.000 subword, cho phép nó biểu diễn một cách linh hoạt các từ trong tiếng Việt, bao gồm cả các từ ghép phức tạp và các từ hiếm gặp.
 - Các Token đặc biệt: Chuỗi token được bổ sung các token đặc biệt. Token <s> được thêm vào đầu chuỗi để biểu thị sự bắt đầu. Token </s> được dùng để phân tách giữa các câu (nếu có) và đánh dấu sự kết thúc của chuỗi.

⁹ <https://ceur-ws.org/Vol-3026/paper25.pdf>

- Xây dựng Vector nhúng
 - **Token Embedding:** Mỗi token (hoặc subword) từ bộ từ vựng BPE được ánh xạ tới một vector nhúng tương ứng. Mỗi token sau đó sẽ được cộng với các vector vị trí và phân đoạn để tạo ra embedding cuối cùng. Embedding này sẽ được đưa vào các cơ chế self-attention trong BERT để thêm thông tin ngữ cảnh cho từng token.
 - **Position Embedding:** Để cung cấp cho mô hình thông tin về vị trí của các token trong chuỗi, một vector nhúng vị trí (position embedding) được cộng vào vector nhúng của token. Các vector vị trí này cũng được học trong quá trình huấn luyện và cho phép mô hình phân biệt giữa các từ giống nhau nhưng xuất hiện ở những vị trí khác nhau. PhoBERT hỗ trợ độ dài chuỗi tối đa là 256 token.
 - **Segment Embedding:** Do mô hình được tối ưu hóa và không sử dụng nhiệm vụ dự đoán câu tiếp theo (Next Sentence Prediction), nó không cần một cơ chế để phân biệt giữa các phân đoạn câu khác nhau. Toàn bộ chuỗi đầu vào được coi là một phân đoạn duy nhất, giúp đơn giản hóa cấu trúc biểu diễn đầu vào.

2.4.4. *Mask Language Model*

Masked Language Model (MLM) là nhiệm vụ tiền huấn luyện (pre-training) cốt lõi và duy nhất của PhoBERT. Kế thừa phương pháp từ RoBERTa, PhoBERT đã loại bỏ nhiệm vụ Dự đoán Câu Tiếp theo (Next Sentence Prediction - NSP) và tập trung hoàn toàn vào MLM để học các biểu diễn ngôn ngữ sâu sắc và mạnh mẽ cho tiếng Việt. Mục tiêu của MLM là huấn luyện mô hình có khả năng dự đoán một từ (token) bị che đi dựa vào ngữ cảnh xung quanh nó.

Bản chất của Masked Language Model (MLM) là một bài tập "điền vào chỗ trống" được thực hiện theo phương pháp tự giám sát (self-supervised). Thông qua cơ chế này, mô hình được huấn luyện bằng cách che đi một số từ ngẫu nhiên trong một câu và sau đó buộc nó phải dự đoán chính xác những từ đã bị che. Để có thể hoàn thành nhiệm vụ này, mô hình không thể chỉ nhìn vào các từ đứng trước hoặc các từ đứng sau một cách riêng lẻ, mà phải học cách hiểu mối quan hệ qua lại giữa tất cả các từ trong toàn bộ câu. Chính quá trình này đã ép mô hình phải xây dựng được các biểu diễn vector hai chiều (bidirectional representations) thực sự sâu sắc và giàu thông tin ngữ cảnh. Đồng thời, một trong những ưu điểm lớn nhất của MLM là khả năng tự tạo dữ liệu huấn luyện. Phương pháp này không đòi hỏi bất kỳ sự gán nhãn thủ công nào; chúng ta có thể tận dụng các kho văn bản thô khổng lồ, tự động che từ để tạo ra vô số cặp dữ liệu gồm đầu vào (câu bị che) và nhãn (từ gốc), giúp quá trình huấn luyện trở nên hiệu quả và có khả năng mở rộng trên quy mô lớn.

Điểm cải tiến quan trọng mà PhoBERT kế thừa từ RoBERTa là việc sử dụng Dynamic Masking (Che động) thay vì Static Masking (Che tĩnh) của BERT gốc:

- Static Masking: Dữ liệu được tiền xử lý một lần duy nhất. Các từ trong câu được che một cách ngẫu nhiên và lưu lại. Trong suốt quá trình huấn luyện, mỗi khi mô hình nhìn thấy một câu cụ thể, câu đó luôn bị che ở những vị trí y hệt.
- Dynamic Masking: Quá trình che từ được thực hiện "động" ngay tại thời điểm dữ liệu được đưa vào mô hình trong mỗi vòng lặp huấn luyện (epoch). Điều này có nghĩa là, cùng một câu gốc có thể được đưa vào mô hình nhiều lần trong các epoch khác nhau, và mỗi lần như vậy, một tập hợp các từ khác nhau sẽ được chọn để che.

Quy trình của Dynamic Masking diễn ra như sau:

1. Chọn câu đầu vào: Một câu được lấy từ kho dữ liệu văn bản tiếng Việt
2. Tách từ và tokenize: Câu đầu vào được tách từ và được mã hóa thành chuỗi token bằng Byte-Pair Encoding
3. Che ngẫu nhiên: Khoảng 15% số token trong chuỗi được chọn ngẫu nhiên để che, với mỗi token được chọn, một trong ba hành động sau sẽ xảy ra:
 - 80% khả năng: Token được thay thế bằng token đặc biệt <mask>
 - 10% khả năng: Token được thay thế bằng một token ngẫu nhiên khác
 - 10% khả năng: token sẽ được giữ nguyên không thay đổi
4. Đưa vào mô hình: Chuỗi token đã bị sửa đổi này được đưa vào các lớp encoder của phoBERT
5. Dự đoán: Mô hình phải sử dụng vector đầu ra tại vị trí của các token bị che để dự đoán token gốc trong bộ từ vựng. Hàm mất mát sẽ tính toán sự khác biệt giữa dự đoán của mô hình và token gốc, từ đó cập nhật trọng số của mô hình

PHẦN 3 KIỂM TRA VÀ ĐÁNH GIÁ CÁC MÔ HÌNH

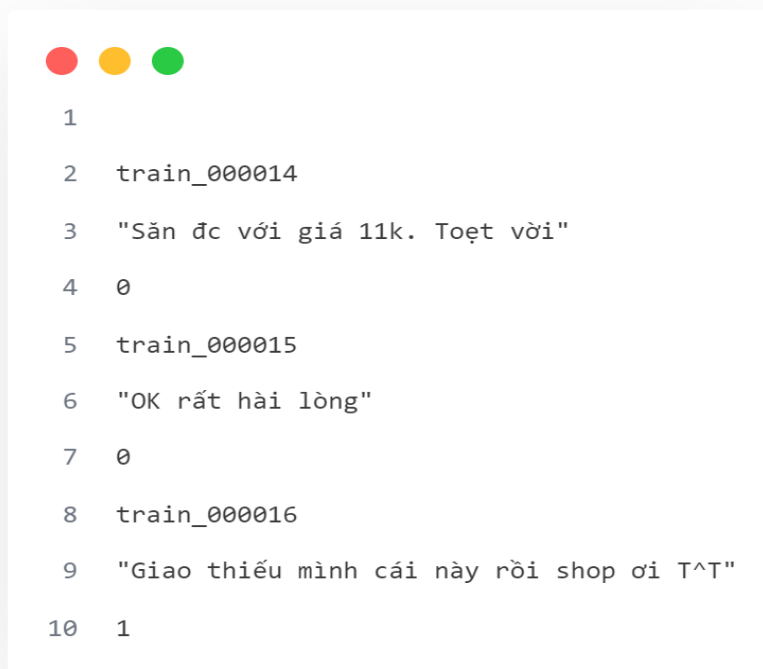
3.1. Xử Lý Dữ Liệu Đầu Vào

3.1.1. Dữ liệu sử dụng

Tập dữ liệu sử dụng gồm training dataset và testing dataset được lấy từ cuộc thi aivivn tổ chức. Trong đó tập training dataset gồm có 16087 câu bình luận đã được gắn nhãn và tập testing có 10981 câu bình luận. Trong tập training, mỗi câu bình luận được định dạng theo thứ tự gồm: mã bình luận, câu bình luận và nhãn của câu bình luận.

Dataset đã được gắn nhãn sẵn, khi quan sát bộ dữ liệu này mình nhận thấy nó khá sạch tức dữ liệu hầu hết đã được gắn nhãn đúng nên mình sẽ không gắn nhãn lại dữ liệu nữa

- Bình luận tích cực, ví dụ : "sản phẩm này đẹp quá", "xuất sắc nha shop", "đẹp lắm ạ" sẽ được gắn nhãn 0
- Bình luận tiêu cực, ví dụ: " sản phẩm không vừa ý", "giao hàng méo mó", "xấu quá" sẽ được gắn nhãn 1



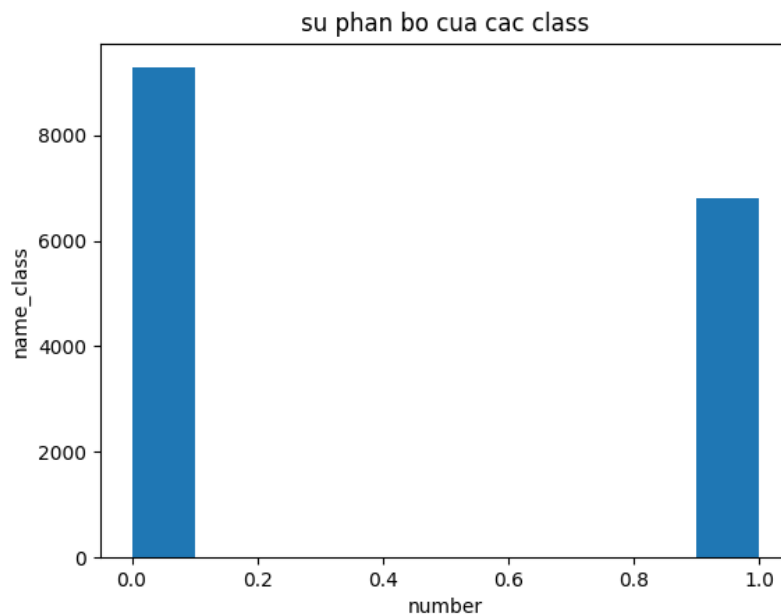
```
1
2  train_000014
3  "Săn đc với giá 11k. Toẹt vời"
4  0
5  train_000015
6  "OK rất hài lòng"
7  0
8  train_000016
9  "Giao thiếu mình cái này rồi shop ời T^T"
10  1
```

Hình 3.1 Dữ liệu đầu vào

3.1.2. So sánh sự phân bố dữ liệu đầu vào

Chúng ta cần phải so sánh sự phân bố của các class trong tập train, vì để đảm bảo sự phân bố giữa các class là cân bằng, nếu quá mất cân bằng thì có thể dẫn đến trường hợp

overfit, khiến kết quả training model bị sai. Ở đây ta sử dụng matplotlib để có thể trực quan hóa dữ liệu dưới dạng biểu đồ.



Hình 3.2 So sánh sự phân bố dữ liệu đầu vào

3.1.3. Đọc và chuyển đổi dữ liệu

Sau khi có được dataset phù hợp, chúng ta bắt đầu đi vào việc chuẩn bị dữ liệu cho việc huấn luyện mô hình. Ta tiến hành đọc dữ liệu từ tập train, tách các mẫu dữ liệu thành từng đoạn, sau đó sẽ chuyển đổi các dữ liệu thô thành dạng dictionary (chứa id, review, label)

Ví dụ: Với input

“ train_001

Sản phẩm này rất tốt

0 “

“ train_002

Giao hàng chậm quá

1 ”

Sau khi xử lý sẽ cho ra kết quả :

[

{'id': 'train_001', 'review': 'Sản phẩm này rất tốt', 'label': 0},

{'id': 'train_002', 'review': 'Giao hàng chậm quá', 'label': 1 }

]

3.1.4. Chuẩn hóa dữ liệu đầu vào

Ta nhận thấy trong tập dữ liệu huấn luyện, có nhiều kiểu bình luận như sau : “sp đẹp quá”, “mon shop”, “thks sôp”, “mk ck r nha”. Đây là những câu bình luận chúng ta có thể thấy với lứa tuổi teen “teen code”. Khi đọc chúng ta có thể hiểu về mặt ý nghĩa như “thks sôp” = “cảm ơn cửa hàng” nhưng máy tính thì không hiểu điều đó, dẫn đến khi chúng ta trích xuất vector cho các câu bình luận có ý nghĩa giống nhau thì sẽ cho chúng ta các vector khác nhau. Vì vậy chúng ta cần chuẩn hóa lại các câu bình luận để có thể trích xuất vector hợp lý hơn. Ở đây, nếu các bạn có thể chuẩn hóa càng nhiều và càng đúng về các bình luận đầu vào thì độ chính xác của model sẽ càng tăng lên

Đoạn code sẽ xử lý, chuẩn hóa các từ teen code thành từ phổ biến, xử lý các câu bình luận rỗng, xử lý các câu bình luận có chữ viết hoa thành viết thường

```
1 from preprocess import DataSource
2 import pandas as pd
3 import re
4 dict = [{"ship", "vận chuyển"}, {"shop", "cửa hàng"}, {"m", "mình"}, {"mik", "mình"}, {"ko", "không"}, {"k", "không"},
5 {"kh", "không"}, {"khong", "không"}, {"kg", "không"}, {"khg", "không"}, {"tl", "trả lời"},
6 {"rep", "trả lời"}, {"r", "rồi"}, {"fb", "facebook"}, {"face", "faceook"}, {"thanks", "cảm ơn"},
7 {"thank", "cảm ơn"}, {"tks", "cảm ơn"}, {"tk", "cảm ơn"}, {"ok", "tốt"}, {"oki", "tốt"}, {"okie", "tốt"}, {"sp", "sản phẩm"},
8 {"dc", "được"}, {"vs", "với"}, {"dt", "điện thoại"}, {"thjk", "thích"}, {"thik", "thích"}, {"qa", "quá"},
9 {"tre", "trẻ"}, {"bgjo", "bao giờ"}, {"h", "giờ"}, {"qa", "quá"}, {"dep", "đẹp"}, {"xau", "xấu"}, {"ib", "nhắn tin"},
10 {"cute", "dễ thương"}, {"sz", "size"}, {"good", "tốt"}, {"god", "tốt"}, {"bt", "bình thường"}]
11 class util():
12     def remove(self, text):
13         if text is None:
14             return "" # Nếu text là None, trả về chuỗi rỗng
15         text = re.sub(r'([A-Z])\1+', lambda m: m.group(1).upper(), text, flags=re.IGNORECASE)
16         return text # Thụt lề đúng
17     def A_cvt_a(self, text): # chuyển các ký tự viết hoa về các ký tự viết thường
18         text = text.lower()
19         return text
20     def utils_data(self, text):
21         list_text = text.split(" ")
22         for i in range(len(list_text)):
23             for j in range(len(dict)):
24                 if (list_text[i] == dict[j][0]):
25                     list_text[i] = dict[j][1]
26         text = " ".join(list_text)
27         return text
28     def text_util_final(self, text):
29         text = self.remove(text)
30         text = self.A_cvt_a(text)
31         text = self.utils_data(text)
32         return text
33
34
35
```

Hình 3.3 Chuẩn hóa dữ liệu đầu vào

3.2. Huấn Luyện Mô Hình TF-IDF Kết Hợp SVM

3.2.1. Trích xuất vector cho model

Sau khi xử lý và chuẩn hóa dữ liệu từ tập dữ liệu huấn luyện, ta bắt đầu xây dựng và chuyển hóa dữ liệu thành vector để phục vụ cho việc huấn luyện mô hình

Ở đây ta sử dụng các hàm có sẵn TfidfVectorizer ở trong thư viện sklearn-feature_extraction.text để chuyển dữ liệu thành vector. Cài đặt số lượng chiều tối đa được lưu trữ trong các vector là 100000, các từ được xét gồm từ đơn, cặp từ và bộ ba từ

3.2.2. Lựa chọn tham số, chia validation

Việc tối ưu các tham số trong TF-IDF và SVM rất quan trọng, nếu tham số không phù hợp với dữ liệu sẽ cho ra kết quả rất thấp và ngược lại, nếu tham số càng phù hợp thì kết quả đạt được của model sẽ càng cao

Ở đây ta sử dụng GridSearch CV¹⁰() được cung cấp sẵn trong Sklearn, hàm sẽ giúp ta đánh giá từng bộ tham số và đưa ra bộ tham số cho kết quả tốt nhất.

Ngoài ra, ta chia tập training thành 5 fold để tạo validation¹¹ cho việc huấn luyện

Sau khi sử dụng GridSearch CV để tìm bộ tham số tốt nhất cho mô hình, ta thu được bộ tham số cho SVM như sau

```
Tham số tốt nhất từ GridSearchCV: {'C': 1, 'kernel': 'linear'},  
Sử dụng C=1, kernel='linear' để huấn luyện model cuối cùng.
```

Hình 3.4 Bộ tham số tối ưu của mô hình SVM

3.2.3. Thử nghiệm model

Sau khi huấn luyện với tập train, mô hình đã được lưu lại để thực hiện phân lớp sau này. Ta gọi lại model để kiểm tra bằng cách nhập các câu bình luận vào và xem kết quả phân lớp của model, ví dụ như sau:

```
--- Phân loại câu ---  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): bộ đồ này đẹp quá  
Dự đoán: Bình luận tích cực!  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): mê quá  
Dự đoán: Bình luận tích cực!  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): đẹp nhưng khá đắt  
Dự đoán: Bình luận tích cực!  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): đắt và xấu quá  
Dự đoán: Bình luận tiêu cực!  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): khá phù hợp với tôi  
Dự đoán: Bình luận tích cực!  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): exit
```

Hình 3.5 Ví dụ về sự phân loại của mô hình SVM

3.3. Huấn Luyện Mô Hình PhoBERT

3.3.1. Xây dựng Tokenizer

Với dữ liệu đầu vào là các câu văn bản đã được chuẩn hóa và làm sạch, ta tiến hành tách từ bằng cách sử dụng Byte-Pair Encoding. Sau đó ta chuyển đổi văn bản thành các

¹⁰ https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

¹¹ https://scikit-learn.org/stable/modules/cross_validation.html

con số (token ID) mà mô hình có thể hiểu. Kết quả thu được là một bộ từ vựng gồm khoảng 64000 subword và các quy tắc để áp dụng BPE

3.3.2. Huấn luyện mô hình với Masked Language Model

Với mô hình đang sử dụng là PhoBERT_BASE, mô hình dùng sẵn tham số được cấu hình với 12 lớp transformer, 768 chiều ẩn, 12 đầu chú ý tương đương với 135 triệu tham số được sử dụng. Chiều dài của thông tin đầu vào được đặt là max_length = 256 trong tokenizer

```
--- Bắt đầu quá trình huấn luyện ---  
Đang tải dữ liệu từ /content/train.crash...  
Số lượng mẫu dữ liệu hợp lệ: 16086  
Khởi tạo Tokenizer và Model BERT...  
Sử dụng thiết bị: cpu  
Đang xử lý văn bản và tạo nhãn...  
Số lượng văn bản để tạo embedding: 16086  
Đang tạo embedding từ BERT cho toàn bộ dữ liệu (có thể mất thời gian)...  
Chia dữ liệu thành tập huấn luyện và tập kiểm tra...  
Số lượng mẫu huấn luyện: 12868, Số lượng mẫu kiểm tra: 3218  
Huấn luyện mô hình SVC...  
Độ chính xác trên tập kiểm tra: 0.8297
```

Hình 3.6 Xây dựng Tokenizer và huấn luyện PhoBERT

3.3.3. Thử nghiệm model

Sau khi huấn luyện với tập train, mô hình đã được lưu lại để thực hiện phân lớp sau này. Ta gọi lại model để kiểm tra bằng cách nhập các câu bình luận vào và xem kết quả phân lớp của model, ví dụ như sau:

```
--- Bắt đầu kiểm tra phân loại câu ---  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): đẹp quá  
Kết quả: Bình luận tích cực!  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): phù hợp với giá tiền  
Kết quả: Bình luận tiêu cực!  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): đắt nhưng đẹp  
Kết quả: Bình luận tích cực!  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): phù hợp với lứa tuổi trẻ trung, shop đỉnh quá  
Kết quả: Bình luận tiêu cực!  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): xấu xí quá  
Kết quả: Bình luận tiêu cực!  
Nhập câu để kiểm tra cảm xúc (hoặc nhập 'exit' để thoát): exit  
--- Chương trình kết thúc ---
```

Hình 3.7 Ví dụ về sự phân loại của mô hình PhoBERT

3.4. Đánh Giá Kết Quả Sau Khi Xây Dựng

3.4.1. Mô hình SVM

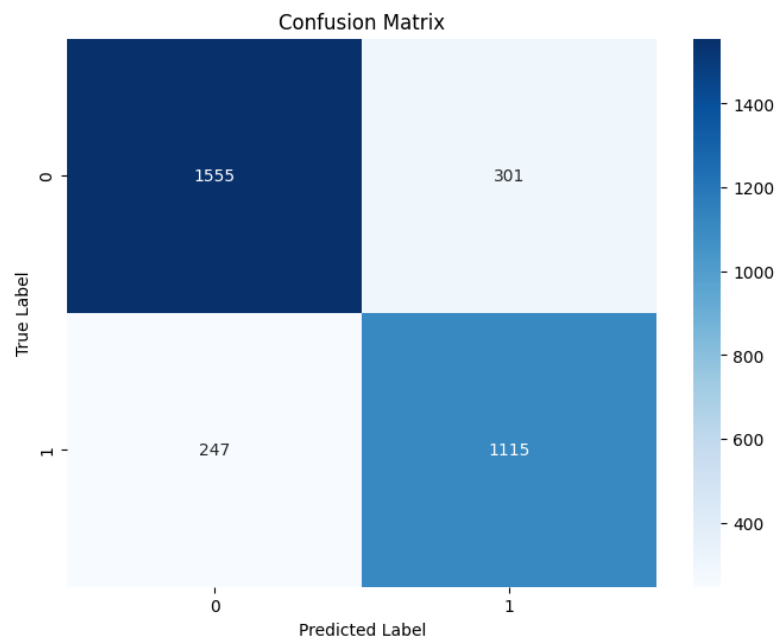
Sau khi huấn luyện mô hình, ta tiến hành đánh giá mô hình với cách chấm f1_score¹²:

¹² <https://www.geeksforgeeks.org/f1-score-in-machine-learning/>

Mô hình đạt được số điểm 80,271% theo tập dữ liệu testing

Model khá nhanh, nhẹ nhàng, dùng tốt với những bình luận thông thường nhưng vẫn còn phân tích sai trong những trường hợp các câu bình luận khó, phức tạp về ý nghĩa

Mặc dù việc dùng Tfidf để trích xuất vector đặc trưng đã có để ý đến thứ tự sắp xếp các từ trong câu nhưng Tfidf vẫn chủ yếu là để đánh trọng số những từ quan trọng là chính nên có thể sai những câu bình luận kiểu như: "Không làm tôi thất vọng" sẽ được xếp vào nhóm tiêu cực. Giải pháp ở đây là ta sẽ nâng cấp cách trích xuất vector để có đầu vào tốt hơn cho thuật toán phân lớp



Hình 3.8 Ma trận nhầm lẫn của mô hình SVM

Các giá trị trong ma trận:

- **True Label 0, Predicted Label 0 (True Negatives - TN): 1555**
 - Đây là số trường hợp thực sự thuộc lớp 0 (ví dụ: "tích cực") và mô hình đã dự đoán chính xác là lớp 0.
- **True Label 0, Predicted Label 1 (False Positives - FP): 301**
 - Đây là số trường hợp thực sự thuộc lớp 0, nhưng mô hình đã dự đoán nhầm là lớp 1 (ví dụ: "tiêu cực"). Đây là lỗi loại I.
- **True Label 1, Predicted Label 0 (False Negatives - FN): 247**
 - Đây là số trường hợp thực sự thuộc lớp 1 (ví dụ: "tiêu cực"), nhưng mô hình đã dự đoán nhầm là lớp 0 (ví dụ: "tích cực"). Đây là lỗi loại II.
- **True Label 1, Predicted Label 1 (True Positives - TP): 1115**

- Đây là số trường hợp thực sự thuộc lớp 1 và mô hình đã dự đoán chính xác là lớp 1.

Nhận xét chung:

Mô hình có khả năng nhận diện tốt cả hai lớp, với số lượng dự đoán đúng cho lớp 0 (TN = 1555) và lớp 1 (TP = 1115) đều cao. Nhìn chung, các giá trị trên đường chéo chính (dự đoán đúng) lớn hơn đáng kể so với các giá trị ngoài đường chéo chính (dự đoán sai), cho thấy mô hình có năng lực phân loại tốt.

- **Lỗi False Positives (FP = 301):** Mô hình đã dự đoán nhầm 301 mẫu từ lớp 0 thành lớp 1. Đây là một con số đáng kể, cho thấy mô hình có xu hướng bị nhầm lẫn và cho rằng một số bình luận "tích cực" là "tiêu cực". Trong thực tế, lỗi này có thể dẫn đến việc phản hồi không phù hợp với các khách hàng đang có trải nghiệm tốt.
- **Lỗi False Negatives (FN = 247):** Mô hình đã bỏ sót 247 trường hợp thực sự thuộc lớp 1. Điều này có nghĩa là một số lượng không nhỏ các bình luận "tiêu cực" đã không được phát hiện và bị phân loại nhầm thành "tích cực". Đây là một lỗi nghiêm trọng đối với các doanh nghiệp muốn chủ động xác định và giải quyết các vấn đề của khách hàng.

Tính toán các độ đo hiệu suất:

Accuracy	0.8297
Recall	0.8186
Precision	0.8378
Specificity	0.7874
F1_score	0.8027

Bảng 3.1 Hiệu suất đánh giá của mô hình SVM

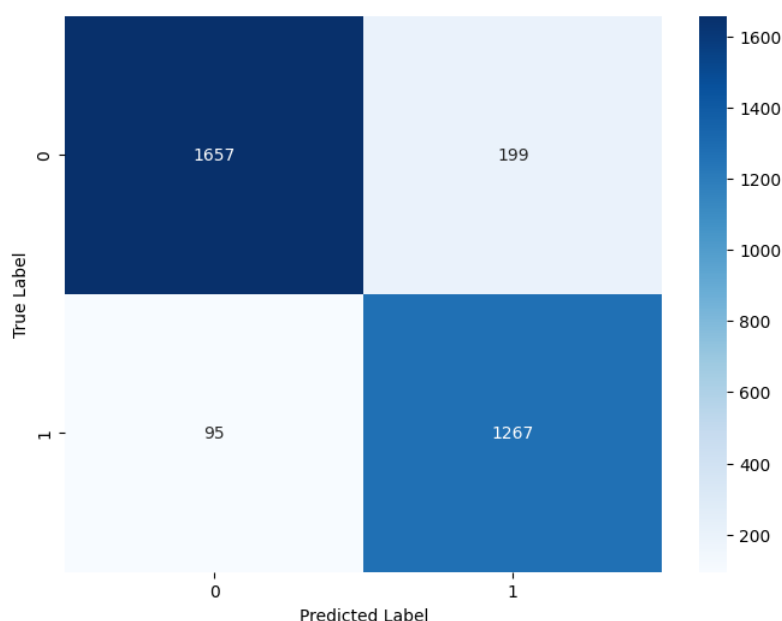
- Đây là một độ chính xác khá tốt, cho thấy mô hình hoạt động hiệu quả trên phần lớn dữ liệu kiểm tra.
- Mô hình có khả năng phát hiện được khoảng 81.86% trong tổng số các trường hợp thực sự thuộc lớp 1.

- Mô hình có khả năng xác định đúng 83.78% các trường hợp thực sự thuộc
- Khi mô hình dự đoán một mẫu là lớp 1, nó chỉ đúng trong khoảng 78.74% trường hợp.

Mô hình SVM với $C=1$ và kernel linear cho thấy hiệu suất tốt trên tập dữ liệu này, với độ chính xác tổng thể cao. Mô hình dường như cân bằng trong việc nhận diện cả hai lớp, mặc dù có một chút xu hướng dự đoán nhầm lớp 0 thành lớp 1 (False Positives) nhiều hơn là dự đoán nhầm lớp 1 thành lớp 0 (False Negatives).

3.4.2. Mô hình PhoBERT

Theo cách chấm điểm `f1_score` dựa trên tập test, mô hình đạt được điểm số là 88,881%



Hình 3.9. Ma trận nhầm lẫn của mô hình PhoBERT

Các giá trị trong ma trận:

- True Label 0, Predicted Label 0 (True Negatives - TN): 1657: Đây là số trường hợp thực sự thuộc lớp 0 (ví dụ: "tích cực") và mô hình đã dự đoán chính xác là lớp 0.
- True Label 0, Predicted Label 1 (False Positives - FP): 199: Đây là số trường hợp thực sự thuộc lớp 0, nhưng mô hình đã dự đoán nhầm là lớp 1 (ví dụ: "tiêu cực"). Đây là lỗi loại I.
- True Label 1, Predicted Label 0 (False Negatives - FN): 95: Đây là số trường hợp thực sự thuộc lớp 1 (ví dụ: "tiêu cực"), nhưng mô hình đã dự đoán nhầm là lớp 0 (ví dụ: "tích cực"). Đây là lỗi loại II.
- True Label 1, Predicted Label 1 (True Positives - TP): 1267: Đây là số trường hợp thực sự thuộc lớp 1 và mô hình đã dự đoán chính xác là lớp 1.

Nhận xét chung:

Mô hình có khả năng nhận diện tốt các mẫu thuộc lớp 0 (TN = 1657). Mô hình cũng có khả năng nhận diện tốt các mẫu thuộc lớp 1 (TP = 1267). Số lượng dự đoán đúng (đường chéo chính) cao hơn đáng kể so với số lượng dự đoán sai (ngoài đường chéo chính).

Lỗi False Positives (FP = 199): Mô hình có xu hướng dự đoán một số mẫu lớp 0 thành lớp 1. Tùy thuộc vào bài toán, lỗi này có thể chấp nhận được hoặc cần cải thiện. Ví dụ, nếu lớp 1 là "spam" và lớp 0 là "không spam", việc một email không spam bị đánh dấu là spam (FP) có thể gây khó chịu cho người dùng.

Lỗi False Negatives (FN = 95): Mô hình ít mắc lỗi này hơn so với False Positives. Điều này có nghĩa là mô hình ít bỏ sót các trường hợp thực sự thuộc lớp 1 hơn. Ví dụ, nếu lớp 1 là "bệnh" và lớp 0 là "không bệnh", việc bỏ sót một trường hợp bệnh (FN) có thể nguy hiểm hơn.

Accuracy	0.9086
Recall	0.9302
Precision	0.8642
Specificity	0.8928
F1_score	0.88881

Bảng 3.2 Hiệu suất đánh giá của mô hình PhoBERT

- Đây là một độ chính xác khá cao, cho thấy mô hình hoạt động tốt trên tập dữ liệu kiểm tra này.
- Mô hình có khả năng phát hiện 93.02% các trường hợp thực sự thuộc lớp 1.
- Mô hình có khả năng xác định đúng 89.28% các trường hợp thực sự thuộc lớp 0.
- Khi mô hình dự đoán một mẫu là lớp 1, nó đúng trong khoảng 86.42% trường hợp.

3.4.3. So sánh giữa hai mô hình

	Accuracy	Recall	Precision	Specificity	F1_score
--	----------	--------	-----------	-------------	----------

TF-IDF + SVM	0.8297	0.8186	0.8378	0.7874	0.8027
PhoBERT	0.9086	0.9302	0.8642	0.8928	0.8888

Bảng 3.3 So sánh hiệu suất của hai mô hình

Mô hình PhoBERT thể hiện sự ưu việt tuyệt đối so với phương pháp TF-IDF + SVM. Độ chính xác tổng thể (Accuracy) của PhoBERT đạt 90.86%, cao hơn đáng kể so với mức 82.97% của TF-IDF + SVM. Tương tự, chỉ số F1-Score, vốn là thước đo cân bằng giữa Precision và Recall, của PhoBERT cũng cao hơn hẳn (88.88% so với 80.27%). Điều này khẳng định rằng PhoBERT không chỉ dự đoán đúng nhiều hơn mà còn duy trì được sự cân bằng tốt trong việc nhận diện các lớp.

Sự chênh lệch lớn về hiệu suất này không phải là ngẫu nhiên mà xuất phát từ sự khác biệt cơ bản về kiến trúc và cách tiếp cận của hai mô hình:

- **Khả năng hiểu ngữ cảnh:** TF-IDF + SVM là một mô hình "túi từ" (bag-of-words), nó chỉ quan tâm đến tần suất xuất hiện của từ mà không hiểu được thứ tự, ngữ nghĩa hay mối quan hệ giữa các từ trong câu. Ngược lại, PhoBERT, với kiến trúc Transformer và cơ chế tự chú ý (self-attention), có khả năng hiểu sâu sắc ngữ cảnh của câu. Nó có thể phân biệt được ý nghĩa của các câu phức tạp, có từ phủ định hoặc cấu trúc đảo ngữ, điều mà TF-IDF + SVM không thể làm được.
- **Kiến thức nền từ tiền huấn luyện:** PhoBERT đã được tiền huấn luyện trên một kho dữ liệu tiếng Việt khổng lồ. Quá trình này giúp nó học được các biểu diễn từ (word embeddings) vô cùng phong phú và giàu ngữ nghĩa. Nó "biết" được các từ đồng nghĩa, trái nghĩa và các mối quan hệ phức tạp khác. Trong khi đó, TF-IDF chỉ đơn thuần là một phương pháp thống kê tần suất từ trên bộ dữ liệu hiện có.

Tóm lại, kết quả thực nghiệm trong Bảng 3.3 đã khẳng định một cách mạnh mẽ rằng mô hình PhoBERT mang lại hiệu quả vượt trội so với phương pháp truyền thống TF-IDF + SVM trong bài toán phân tích sắc thái bình luận tiếng Việt. Sự ưu việt này thể hiện ở tất cả các khía cạnh, từ độ chính xác tổng thể đến khả năng cân bằng giữa Precision và Recall. Điều này chứng tỏ sức mạnh của các mô hình ngôn ngữ lớn dựa trên kiến trúc Transformer trong việc xử lý các tác vụ ngôn ngữ tự nhiên phức tạp.

KẾT LUẬN

Đồ án đã đạt được mục tiêu đề ra

Kết quả của đồ án

1. Trình bày nội dung lý thuyết về xử lý ngôn ngữ tự nhiên, các mô hình học máy và mô hình học sâu
2. Xây dựng được các mô hình học máy, học sâu để phân tích sắc thái bình luận
3. Đồ án đã đưa ra được các kết quả thực nghiệm và đánh giá mô hình sau quá trình xây dựng và tinh chỉnh

Kỹ năng đạt được

1. Đọc và nghiên cứu các báo cáo khoa học chuyên ngành, tài liệu tham khảo uy tín trên thế giới
2. Viết đồ án và báo cáo có trình tự rành mạch, cụ thể
3. Rèn luyện khả năng tư duy xử lý các mô hình
4. Kỹ năng lập trình cũng trở nên thành thạo và tối ưu hơn

Hướng Phát Triển

Dựa trên những kết quả đã đạt được và các hạn chế còn tồn tại của đề tài, chúng tôi đề xuất một số hướng phát triển tiềm năng trong tương lai để nâng cao hiệu quả và tính ứng dụng của mô hình:

- **Mở rộng và nâng cao chất lượng bộ dữ liệu:** Hướng phát triển nền tảng và quan trọng nhất trong tương lai là việc mở rộng và nâng cao chất lượng của bộ dữ liệu huấn luyện.
- **Thử nghiệm và tối ưu hóa các kiến trúc mới:** Trước hết, nghiên cứu sẽ thử nghiệm với phiên bản vinai/phobert-large, một mô hình có số lượng tham số lớn hơn, hứa hẹn khả năng học các biểu diễn ngữ nghĩa sâu sắc hơn.
- **Xây dựng và triển khai thành ứng dụng thực tế:** Mục tiêu cuối cùng của nghiên cứu ứng dụng là chuyển đổi các mô hình đã được huấn luyện thành những công cụ có thể sử dụng được trong thực tế.

TÀI LIỆU THAM KHẢO

- [1] M. M. B. I. Wisam A. Qader, “An Overview of Bag of Words;Importance, Implementation, Applications, and Challenges,” p. June, 2019. https://www.researchgate.net/publication/338511771_An_Overview_of_Ba_g_of_WordsImportance_Implementation_Applications_and_Challenges
- [2] A. T. N. Dat Quoc Nguyen, “PhoBERT: Pre-trained language models for Vietnamese,” 2020. <https://aclanthology.org/2020.findings-emnlp.92.pdf>
- [3] M.-W. C. K. L. K. T. Jacob Devlin, “BERT: Pre-training of Deep Bidirectional Transformers for,” 2019. <https://arxiv.org/pdf/1810.04805>
- [4] B. Smith, “A Complete Guide to BERT with Code,” *towards data science*, 13 May 2024. <https://towardsdatascience.com/a-complete-guide-to-bert-with-code-9f87602e4a11/>
- [5] T. N. Nhat, “Phân tích sắc thái bình luận,” *VietNamLab*, 4 August 2019. <https://blog.vietnamlab.vn/xay-dung-1-model-machine-learning-don-gian-de-giai-quyet-bai-toan-phan-loai-sac-thai-binh-luan-trong-tieng-viet/>
- [6] geeksgorgeeks, “Geeksforgeeks,” Neural-Network [Trực tuyến]. Available: <https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/>.
- [7] scikit-learn, “scikit-learn,” GridSearchCV [Trực tuyến]. Available: https://scikit-learn.org/stable/modules/cross_validation.html
- [8] M. O. N. G. J. D. M. J. D. C. O. L. M. L. L. Z. V. S. Yinhan Liu, “RoBERTa: A Robustly Optimized BERT Pretraining Approach,” số RoBERT, 2019. <https://arxiv.org/abs/1907.11692>
- [9] W. L.Taylor, “Cloze Procedure: A New Tool for Measuring Readability,” *Journalism Quarterly*, pp. 415-433, 1953
<https://gwern.net/doc/psychology/writing/1953-taylor.pdf>
- [10] N. S. N. P. Ashish Vaswani, “The Annotated Transformer,” *Havard nlp*, 3 Apr 2018. <https://nlp.seas.harvard.edu/2018/04/03/attention.html>

