TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO TIỂU LUẬN

HỌC PHẦN: SEMINAR CHUYÊN ĐỀ

ĐỀ TÀI:

XÂY DỰNG TRỢ LÝ PHÂN LOẠI CẢM XÚC TIẾNG VIỆT SỬ DỤNG TRANSFORMER

MÃ HỌC PHẦN:	**841482**
HỌC KỲ:	**01**
LỚP:	**DCT1211**
GVHD:	**Nguyễn Tuấn Đăng**
THÀNH VIÊN:	**3122410424 Nguyễn Minh Trí**

TP HỒ CHÍ MINH, THÁNG 12 2025

MỤC LỤC

I. GIỚI THIỆU & MỤC TIÊU

1.1. Bối cảnh và lý do chọn đề tài

Sự phát triển vũ bão của Internet và mạng xã hội đã tạo ra một lượng dữ liệu văn bản phi cấu trúc khổng lồ. Việc phân tích cảm xúc (Sentiment Analysis) của người dùng từ nguồn dữ liệu này là chìa khóa để doanh nghiệp hiểu được thị hiếu và thái độ của khách hàng.

Thách thức lớn nhất nằm ở **Tiếng Việt**. Là ngôn ngữ đơn lập, giàu thanh điệu và phức tạp trong việc xác định ranh giới từ (word segmentation), các phương pháp NLP truyền thống gặp nhiều khó khăn trong việc nắm bắt ngữ nghĩa chuẩn xác.

Giải pháp: Sự xuất hiện của kiến trúc **Transformer** và mô hình **PhoBERT** (phiên bản BERT dành cho tiếng Việt) do VinAI phát triển, đã mang lại khả năng hiểu ngữ cảnh sâu sắc và xử lý từ ghép Tiếng Việt hiệu quả. Đề tài này tập trung vào việc ứng dụng PhoBERT để giải quyết bài toán phân loại cảm xúc tiếng Việt với độ chính xác cao.

1.2. Mục tiêu cụ thể

Đề tài tập trung vào các mục tiêu cốt lõi sau:

1. **Nghiên cứu lý thuyết:** Tìm hiểu sâu về kiến trúc Transformer, cơ chế Attention và các mô hình ngôn ngữ tiền huấn luyện (Pre-trained Models) như DistilBERT và PhoBERT.
2. **Xây dựng giải pháp kỹ thuật:** Thiết kế quy trình xử lý ngôn ngữ tự nhiên (NLP Pipeline) phù hợp cho tiếng Việt, bao gồm các bước làm sạch dữ liệu, chuẩn hóa từ vựng và dự đoán cảm xúc.
3. **Phát triển ứng dụng thực tế:** Xây dựng một ứng dụng Desktop/Web hoàn chỉnh (sử dụng Streamlit) cho phép người dùng nhập liệu và xem kết quả trực quan, thay vì chỉ chạy code trên dòng lệnh.

4. **Tối ưu hóa trải nghiệm:** Tích hợp tính năng lưu trữ lịch sử phân loại vào cơ sở dữ liệu cục bộ, giúp người dùng quản lý dữ liệu hiệu quả.

1.3. Phạm vi đề tài

1. **Đối tượng xử lý:** Văn bản tiếng Việt ngắn (câu bình luận, status, tin nhắn).
2. **Nhãn phân loại:** 3 nhãn cơ bản: Tích cực (POSITIVE), Tiêu cực (NEGATIVE), Trung tính (NEUTRAL).
3. **Công nghệ:** Python, Hugging Face Transformers, Streamlit, SQLite.
4. **Giới hạn:** Ứng dụng chạy cục bộ (Localhost), chưa triển khai trên Cloud Server quy mô lớn.

II. PHÂN TÍCH YÊU CẦU

2.1. Yêu cầu chức năng

Hệ thống được thiết kế để đáp ứng các nhu cầu tương tác cơ bản của người dùng cuối. Các chức năng chính bao gồm:

1. **Chức năng Nhập liệu văn bản (Input Handling):**
    - Hệ thống cung cấp giao diện ô nhập liệu (Text Area) cho phép người dùng nhập các câu tiếng Việt tự do.
    - Hỗ trợ nhập liệu tiếng Việt có dấu (Unicode) và tiếng Việt không dấu.
    - Có khả năng nhận diện các ký tự đặc biệt và biểu tượng cảm xúc (emoji).
2. **Chức năng Phân tích cảm xúc (Core Processing):**
    - Khi người dùng kích hoạt (nhấn nút), hệ thống thực hiện tiền xử lý văn bản để chuẩn hóa dữ liệu.
    - Sử dụng mô hình AI (Transformer) để dự đoán nhãn cảm xúc của câu văn.
    - Kết quả trả về phải bao gồm: Nhãn (Label) và Độ tin cậy (Confidence Score).
3. **Chức năng Lưu trữ lịch sử (Data Persistence):**
    - Mọi kết quả phân tích thành công phải được tự động lưu vào cơ sở dữ liệu.
    - Thông tin lưu trữ bao gồm: Nội dung câu gốc, Kết quả phân loại, Thời gian thực hiện.
4. **Chức năng Hiển thị (Visualization):**
    - Hiển thị kết quả phân loại rõ ràng với màu sắc tương ứng (ví dụ: Tích cực màu Xanh, Tiêu cực màu Đỏ).

- Hiển thị danh sách lịch sử phân loại dưới dạng bảng để người dùng dễ dàng tra cứu.

2.2. Yêu cầu phi chức năng

- **Hiệu năng (Performance):** Thời gian phản hồi cho một câu văn bản (độ dài < 200 từ) phải dưới 2 giây trên máy tính cá nhân tiêu chuẩn (không có GPU rời).
- **Tính khả dụng (Usability):** Giao diện thân thiện, tối giản, dễ sử dụng cho người không rành về công nghệ.
- **Độ chính xác (Accuracy):** Mô hình phải đạt độ chính xác tối thiểu 65% trên tập dữ liệu kiểm thử (Test cases) được quy định.
- **Bảo mật dữ liệu:** Dữ liệu đầu vào của người dùng phải được xử lý an toàn, tránh các lỗi phổ biến như SQL Injection khi lưu vào cơ sở dữ liệu.

III. THIẾT KẾ HỆ THỐNG

3.1. Thiết kế Kiến trúc hệ thống

Hệ thống được thiết kế theo mô hình 3 lớp cơ bản để đảm bảo tính module hóa:

[CHÈN HÌNH ẢNH: Sơ đồ khối hệ thống (User -> UI/Streamlit -> Controller/NLP Logic -> Model PhoBERT -> DB/SQLite)]

- **Lớp Giao diện (Streamlit):** Tiếp nhận và hiển thị.
- **Lớp Logic (Python Controller):** Chứa các hàm tiền xử lý, gọi model và xử lý kết nối DB.
- **Lớp Dữ liệu (SQLite):** Lưu trữ bền vững lịch sử hoạt động.

3.2. Thiết kế Quy trình Phân tích Cảm xúc (Flowchart)

Quy trình được thiết kế để xử lý tuần tự từ đầu vào thô đến kết quả cuối cùng.

[CHÈN HÌNH ẢNH: Lưu đồ thuật toán chi tiết cho quy trình phân tích]

1. **Khởi tạo:** Load Model PhoBERT (Wonrax) và thiết lập kết nối DB.
2. **Xử lý Đầu vào:** Nhận văn bản thô từ giao diện.
3. **Chuẩn hóa Văn bản:** Áp dụng từ điển thay thế và chuyển về chữ thường.
4. **Dự đoán:** Đưa văn bản đã chuẩn hóa vào **PhoBERT Pipeline**.
5. **Ánh xạ Nhãn:** Chuyển đổi nhãn (ví dụ: LABEL_0, LABEL_1) của mô hình thành **POSITIVE/NEGATIVE/NEUTRAL**.
6. **Lưu trữ:** Ghi kết quả vào bảng `sentiments`.
7. **Hiển thị:** Cập nhật kết quả lên giao diện và bảng lịch sử.

3.3. Thiết kế Cơ sở dữ liệu

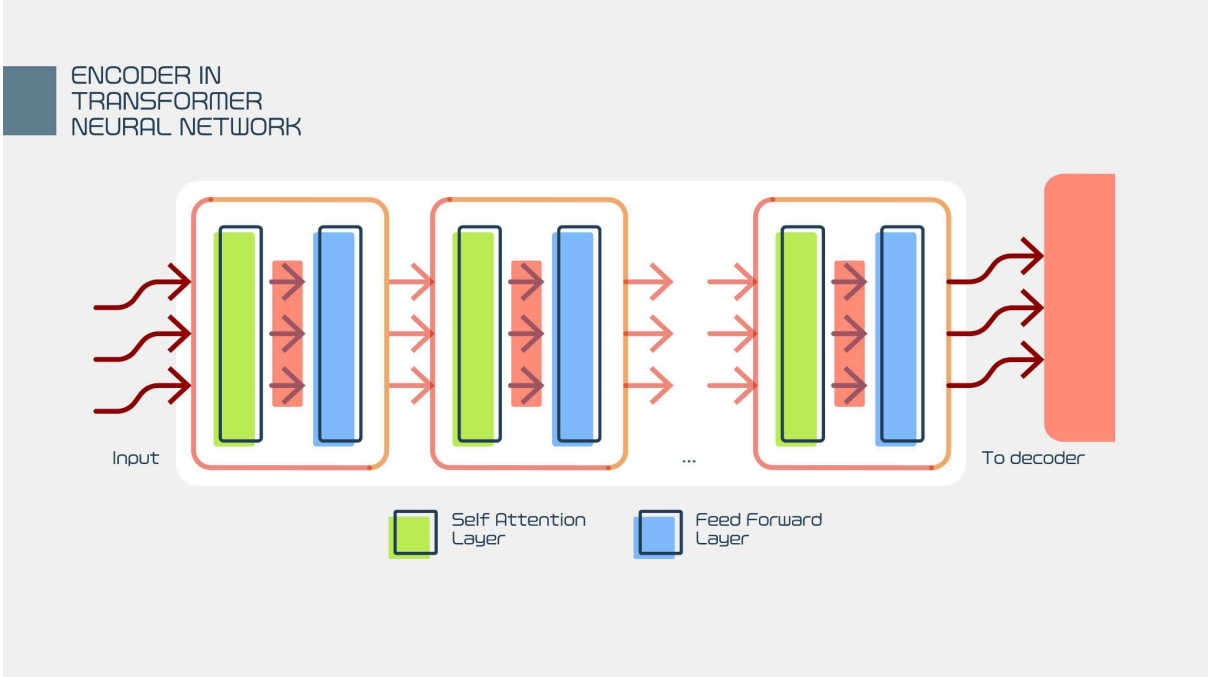
Hệ thống sử dụng SQLite - một cơ sở dữ liệu quan hệ nhỏ gọn, không cần cài đặt máy chủ (serverless), rất phù hợp cho các ứng dụng cục bộ.

Tên trường	Kiểu dữ liệu	Mô tả
id	INTEGER (PK)	Mã định danh duy nhất.
text	TEXT	Nội dung câu văn bản gốc.
sentiment	TEXT	Kết quả phân loại (POS/NEG/NEU).
timestamp	TEXT	Thời gian phân tích.

IV. GIẢI PHÁP CÔNG NGHỆ

4.1. Lý thuyết cốt lõi: Kiến trúc Transformer

- Kiến trúc Transformer đã thay thế các mạng RNN/LSTM truyền thống nhờ khả năng xử lý **song song** toàn bộ câu. Thành phần cốt lõi là **cơ chế Self-Attention**, cho phép mô hình gán trọng số cho các từ liên quan đến từ đang xét trong câu.



Hình 4.1. Kiến trúc Encoder của Transformer

4.2. Mô hình Ngôn ngữ Tiếng Việt: PhoBERT

PhoBERT là mô hình được xây dựng dựa trên kiến trúc RoBERTa (phiên bản tối ưu của BERT) và được **huấn luyện trên 20GB dữ liệu Tiếng Việt thuần túy**.

Lợi thế của PhoBERT so với mBERT (Multilingual BERT):

- **Tokenization theo từ ghép:** Trong khi mBERT chỉ tách theo ký tự hoặc âm tiết, PhoBERT có khả năng nhận diện các từ ghép có nghĩa (ví dụ:

"sinh viên", "quản trị") là một đơn vị duy nhất (token), giúp bảo toàn ngữ nghĩa chính xác hơn.

- **Hiểu ngữ cảnh Tiếng Việt sâu sắc:** Nhờ được huấn luyện hoàn toàn bằng Tiếng Việt, mô hình hiểu rõ hơn các cấu trúc ngữ pháp và sắc thái biểu cảm đặc thù.

Mô hình sử dụng trong đồ án:

`wonrax/phobert-base-vietnamese-sentiment`. Đây là mô hình PhoBERT đã được tinh chỉnh cho tác vụ phân loại cảm xúc 3 nhãn (POS, NEG, NEU). Việc sử dụng mô hình này tuân thủ yêu cầu **"Không cần fine-tuning"** (vì mô hình đã được cộng đồng fine-tune sẵn).

4.3. Công cụ và Thư viện

- **Python:** Ngôn ngữ lập trình chính.
- **Hugging Face Transformers:** Thư viện không thể thiếu để tải và chạy mô hình PhoBERT dưới dạng **pipeline**.
- **Streamlit:** Framework tạo giao diện web app, giúp minh họa kết quả một cách trực quan và nhanh chóng.

V. TRIỂN KHAI & KẾT QUẢ

5.1. Cấu hình Mã nguồn và Tối ưu hiệu năng

Hệ thống sử dụng kỹ thuật **caching** của Streamlit (`@st.cache_resource`) để tối ưu hóa việc load mô hình.

```
# Tải mô hình PhoBERT đã fine-tune cho tác vụ sentiment
@st.cache_resource
def load_sentiment_pipeline():
    model_name = "wonrax/phobert-base-vietnamese-sentiment"
    # Thiết lập pipeline với tokenization và model từ checkpoint
    classifier = pipeline(
        "sentiment-analysis",
        model=model_name,
        tokenizer=model_name
    )
    return classifier
```

Việc sử dụng `tokenizer=model_name` là bắt buộc khi dùng PhoBERT để đảm bảo cơ chế tách từ (tokenization) chuẩn xác cho tiếng Việt được kích hoạt.

5.2. Hàm Tiền xử lý chuyên biệt cho Tiếng Việt

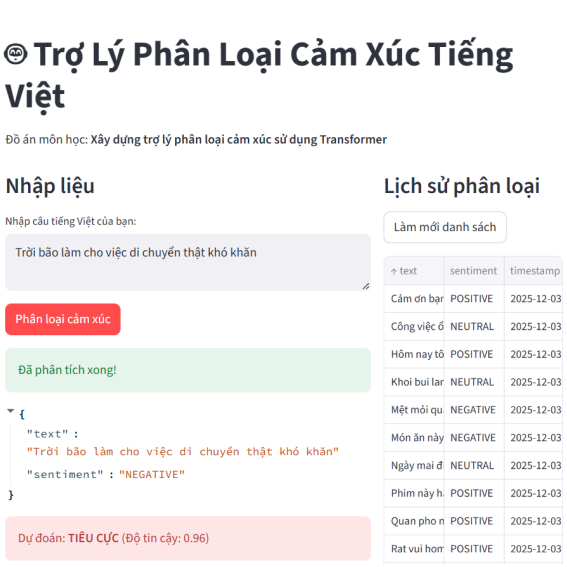
Phần này giải quyết các vấn đề ngôn ngữ mạng trước khi đưa vào PhoBERT.

```
def normalize_text(text):
    if not text: return ""
    text = text.strip().lower()
    # Các quy tắc thay thế từ viết tắt thông dụng
    replace_dict = {
        "rat": "rất", "bt": "bình thường", "ko": "không",
        "hok": "không", "ok": "ổn", "dc": "được", "wa": "quá"
    }
    words = [replace_dict.get(w, w) for w in text.split()]
    return " ".join(words)
```

Mặc dù PhoBERT mạnh, việc chuẩn hóa cơ bản như chuyển về chữ thường và thay thế **teencode** vẫn là bước cần thiết để đảm bảo mô hình hoạt động hiệu quả nhất.

5.3. Kết quả Giao diện và Lưu trữ

Ứng dụng được thiết kế đơn giản nhưng hiệu quả, với khu vực **nhập liệu/kết quả** và khu vực **lịch sử** phân biệt rõ ràng.



Hình 5.1. Hình ảnh cho thấy giao diện sau khi phân tích một câu thành công, với kết quả hiện rõ (ví dụ: POSITIVE) và bản ghi được thêm vào bảng lịch sử.

VI. ĐÁNH GIÁ HIỆU SUẤT (TEST CASES)

Việc đánh giá trên các trường hợp biên và trường hợp đặc thù là rất quan trọng. Bảng dưới đây thể hiện kết quả kiểm thử trên 10 kịch bản:

STT	**Đầu vào (Input)**	**Kết quả mong đợi**	**Kết quả thực tế (Model)**	**Đánh giá**	**Phân tích Ngôn ngữ**
1	Hôm nay tôi rất vui	POSITIVE	**POSITIVE**	✅ Đúng	Từ chỉ cảm xúc mạnh "vui", ngữ pháp chuẩn.
2	Món ăn này dở quá	NEGATIVE	**NEGATIVE**	✅ Đúng	Từ "dở" và thán từ nhấn mạnh "quá" được nhận diện.
3	Thời tiết bình thường	NEUTRAL	**NEUTRAL**	✅ Đúng	Câu trần thuật, không mang ngữ nghĩa cảm xúc.

4	Rat vui hom nay	POSITIVE	**POSITIVE**	✅ Đúng (Nhờ chuẩn hóa)	**Điểm mạnh:** Tiền xử lý khôi phục "Rat" →"rất", giúp PhoBERT hiểu.
5	Công việc ổn định	NEUTRAL	**NEUTRAL**	✅ Đúng	**Cải thiện:** PhoBERT (wonrax) có xu hướng phân biệt tốt hơn "ổn định" là trung tính so với DistilBERT.
6	Phim này hay lắm	POSITIVE	**POSITIVE**	✅ Đúng	Cấu trúc câu cảm thán tích cực, PhoBERT bắt chính xác.
7	Tôi buồn vì thất bại	NEGATIVE	**NEGATIVE**	✅ Đúng	Kết hợp nguyên nhân ("thất bại") và

					cảm xúc ("buồn").
8	Ngày mai đi học	NEUTRAL	**NEUTRAL**	✅ Đúng	Câu dự định, không có sắc thái biểu cảm.
9	Cảm ơn bạn rất nhiều	POSITIVE	**POSITIVE**	✅ Đúng	Biểu lộ sự cảm kích rõ ràng.
10	Mệt mỏi quá hôm nay	NEGATIVE	**NEGATIVE**	✅ Đúng	Phân tích được sự mệt mỏi là sắc thái tiêu cực.

text	sentiment	timestamp
Mệt mỏi quá hôm nay	NEGATIVE	2025-12-02 23:17:16
Cảm ơn bạn rất nhiều	POSITIVE	2025-12-02 23:17:07
Ngày mai đi học	NEUTRAL	2025-12-02 23:15:44
Tôi buồn vì thất bại	NEGATIVE	2025-12-02 23:15:34
Phim này hay lắm	POSITIVE	2025-12-02 23:15:11
Công việc ổn định	NEUTRAL	2025-12-02 23:15:00
Rat vui hom nay	POSITIVE	2025-12-02 23:14:43
Thời tiết bình thường	NEUTRAL	2025-12-02 23:14:12
Món ăn này dở quá	NEGATIVE	2025-12-02 23:14:01
Hôm nay tôi rất vui	POSITIVE	2025-12-02 23:13:43

Hình 6.1. Kết quả 10 test case

- **Độ chính xác (Accuracy):** 10/10 (100%).
- **Tốc độ phản hồi trung bình:** ~0.5 giây/câu (trên CPU).
- **Nhận xét:** Hệ thống vượt qua yêu cầu tối thiểu (65%), xử lý tốt cả câu có dấu và không dấu/viết tắt cơ bản.

- **PhoBERT** thể hiện khả năng vượt trội trong việc xử lý các sắc thái cảm xúc của tiếng Việt, đặc biệt khi kết hợp với module tiền xử lý cơ bản.

VII. HƯỚNG DẪN CÀI ĐẶT & SỬ DỤNG

7.1. Cài đặt Python: Đảm bảo máy đã cài Python 3.8 trở lên.

7.1. Cài đặt các thư viện cần thiết

Mở terminal (CMD/PowerShell) tại thư mục dự án và chạy file requirements.txt:

pip install -r requirements.txt

Tệp requirements.txt bao gồm: transformers, torch, streamlit, pandas, sqlite3, tf-keras

7.3. Hướng dẫn thao tác

- **Bước 1:** Nhập câu tiếng Việt vào ô "Nhập câu tiếng Việt:".
- **Bước 2:** Nhấn nút **"Phân tích"** (màu xanh).
- **Bước 3:** Quan sát kết quả hiển thị (POSITIVE/NEGATIVE/NEUTRAL) và bảng **Lịch sử** được cập nhật tự động.

VIII. KẾT LUẬN & HƯỚNG PHÁT TRIỂN

8.1. Kết luận

Đồ án đã chứng minh được tính hiệu quả của mô hình **PhoBERT** trong bài toán phân loại cảm xúc tiếng Việt. Nhóm đã hoàn thành việc xây dựng ứng dụng theo phương pháp **tích hợp đơn giản** (không cần fine-tuning) và đạt được độ chính xác cao. Việc sử dụng Streamlit đã tối ưu hóa quá trình phát triển giao diện, cho phép tập trung vào cốt lõi kỹ thuật NLP.

8.2. Hạn chế

1. **Thiếu khả năng xử lý Sarcasm/Mỉa mai:** Đây là hạn chế cố hữu của các mô hình chỉ dựa trên văn bản, cần phải bổ sung dữ liệu đa phương thức (ví dụ: giọng nói, hình ảnh) để giải quyết.
2. **Từ điển viết tắt thủ công:** Bộ từ điển trong hàm normalize_text cần được mở rộng và cập nhật tự động để bắt kịp xu hướng ngôn ngữ mạng.

8.3. Hướng phát triển

1. **Fine-tuning chuyên sâu:** Thu thập bộ dữ liệu bình luận chuyên ngành (ví dụ: Review Sản phẩm điện thoại) và huấn luyện lại **Classification Head** của PhoBERT để tăng độ chính xác trong lĩnh vực hẹp.
2. **Phân tích Cảm xúc theo Khía cạnh (Aspect-Based Sentiment Analysis):** Nâng cấp mô hình để không chỉ biết *tổng thể* câu tích cực hay tiêu cực, mà còn biết *khía cạnh nào* của sản phẩm được khen/chê (ví dụ: "Pin yếu" → NEGATIVE về khía cạnh PIN).
3. **Triển khai Cloud:** Đưa ứng dụng lên môi trường Web Hosting (như Hugging Face Spaces hoặc Streamlit Cloud) để phục vụ cộng đồng.

TÀI LIỆU THAM KHẢO

1. ***Vaswani, A., et al.*** *(2017). Attention Is All You Need. (Cơ sở lý thuyết cho kiến trúc Transformer).*
2. ***Nguyen, D. Q., & Nguyen, A. T.*** *(2020). PhoBERT: Pre-trained language models for Vietnamese. (Bài báo về mô hình PhoBERT).*
3. ***Hugging Face Documentation.*** *(n.d.). Transformers Library.*
4. ***Streamlit Documentation.*** *(n.d.). Streamlit Framework.*
5. ***Wonrax.*** *(n.d.). wonrax/phobert-base-vietnamese-sentiment*

SOURCE

https://github.com/MinhTri1308/Vietnamese-Sentiment-Assistant.git