

ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC



**ỨNG DỤNG CHATGPT TRONG BÀI TOÁN  
PHÂN TÍCH SẮC THÁI BÌNH LUẬN SÁCH**

**ĐỒ ÁN I**

**Chuyên ngành: Toán - Tin**

**Chuyên sâu: Khai phá dữ liệu**

**Giảng viên hướng dẫn: TS Trần Ngọc Thăng**

**Sinh viên thực hiện: Nguyễn Quang Minh - 20206157**

**Lớp: Toán tin 01 – K65**

**Hà Nội – 2023**

# Nhận xét của giảng viên hướng dẫn

## 1. Mục tiêu và nội dung đề án

- Hiểu rõ khả năng của ChatGPT trong việc đánh nhãn dữ liệu.
- Phát biểu và nhận dạng các loại bài toán phân tích sắc thái.
- Nắm rõ chiến lược, quy trình đánh nhãn dữ liệu bằng ChatGPT.
- Xây dựng được các mô hình Machine Learning, Deep Learning áp dụng vào bài toán phân tích cảm xúc văn bản.

## 2. Kết quả đạt được

Sau khi hoàn thành đề án, em đã đạt được những kết quả sau:

- Biết cách sử dụng API key, selenium để tự động hóa quá trình gán nhãn dữ liệu bằng ChatGPT.
- Biết cách sử dụng selenium để crawl dữ liệu từ web.
- Nắm rõ được ý tưởng đằng sau các thuật toán học máy (SVM), học sâu (LSTM, PhoBERT) để áp dụng vào mô hình phân loại.
- Triển khai được mô hình SVM, LSTM, PhoBERT vào bài toán phân tích sắc thái văn bản với bộ dữ liệu về bình luận sách với độ chính xác khá tốt.

## 3. Ý thức làm việc của sinh viên

*Hà Nội, ngày 27 tháng 7 năm 2023*

**Giảng viên hướng dẫn**

**TS. Trần Ngọc Thăng**

**MỤC LỤC**

<b>MỞ ĐẦU .....</b>	<b>5</b>
<b>Chương I: Tổng quan về ChatGPT .....</b>	<b>6</b>
<b>1. Giới thiệu chung.....</b>	<b>6</b>
<b>2. Kiến trúc ChatGPT .....</b>	<b>6</b>
<b>3. Đặc điểm của ChatGPT.....</b>	<b>12</b>
<b>Chương II: Bài toán phân tích sắc thái.....</b>	<b>15</b>
<b>1. Phân tích sắc thái cấp độ tài liệu (SD) .....</b>	<b>18</b>
<b>2. Phân tích sắc thái cấp độ câu (SC) .....</b>	<b>20</b>
<b>3. Phân tích sắc thái theo khía cạnh (ABSA) .....</b>	<b>21</b>
<b>Chương III: ChatGPT và việc đánh nhãn dữ liệu .....</b>	<b>23</b>
<b>1. Khảo sát khả năng đánh nhãn dữ liệu của ChatGPT qua các test case .....</b>	<b>23</b>
<b>2. Chiến lược.....</b>	<b>27</b>
<b>3. Quy trình .....</b>	<b>27</b>
<b>4. Kỹ thuật .....</b>	<b>29</b>
<b>Chương IV: Phương pháp giải quyết bài toán .....</b>	<b>39</b>
<b>1. Động lực lựa chọn phương pháp giải quyết .....</b>	<b>39</b>
<b>2. Cơ sở lý thuyết áp dụng cho bài toán.....</b>	<b>40</b>
<b>2.1. TF-IDF (Term Frequency - Inverse Document Frequency) .....</b>	<b>40</b>
<b>2.2. SVM (Support Vector Machine) .....</b>	<b>42</b>
<b>2.3. LSTM (Long Short Term Memory) .....</b>	<b>43</b>

2.4. PhoBERT .....	49
<b>Chương V: Các bước giải quyết bài toán.....</b>	<b>52</b>
1. Crawl dữ liệu .....	52
2. Tổng quan về bộ dữ liệu .....	54
3. Tiền xử lý dữ liệu .....	56
4. Xây dựng mô hình.....	58
4.1. SVM .....	58
4.2. LSTM.....	58
4.3. PhoBERT .....	61
5. Đánh giá kết quả .....	66
<b>KẾT LUẬN .....</b>	<b>67</b>
<b>TÀI LIỆU THAM KHẢO .....</b>	<b>68</b>

## MỞ ĐẦU

Trong thời đại công nghệ 4.0 ngày nay khi mà các trang web, mạng xã hội phát triển nhanh chóng, xuất hiện rất nhiều các câu bình luận của con người về một sản phẩm thuộc một lĩnh vực nào đó, hay một chủ đề hot trên mạng xã hội,... Việc phân tích cảm xúc của người dùng trong những câu bình luận trên là rất quan trọng và cần thiết đối với chúng ta. Các doanh nghiệp, tổ chức có thể theo dõi được phản hồi của khách hàng về dịch vụ, sản phẩm của mình để có thể đáp ứng được nhu cầu của khách hàng. Bằng những công cụ thu thập dữ liệu hiện nay thì nguồn tài nguyên đó có thể được lấy về một cách dễ dàng, nhưng việc đánh nhãn dữ liệu thì vẫn còn là một vấn đề nan giải khi mà phải cần đội ngũ nhưng chuyên gia, nguồn lực về con người để đánh nhãn dữ liệu một cách thủ công, ... Vậy nên ChatGPT đã ra đời để giúp chúng ta có thể giải quyết được vấn đề đó.

# Chương I: Tổng quan về ChatGPT

## 1. Giới thiệu chung

ChatGPT là một mô hình ngôn ngữ lớn được xây dựng dựa trên kiến trúc GPT 3.5 (Generative Pre-training Transformer) được phát triển bởi OpenAI. Mô hình GPT được coi là bước đột phá lớn nhất trong lĩnh vực xử lý ngôn ngữ tự nhiên hiện nay.

ChatGPT có khả năng hiểu và tạo ra được ngôn ngữ tự nhiên. Nó cho phép người dùng đặt câu hỏi và chatbot này sẽ tự động tạo ra các câu trả lời phù hợp nhất dưới dạng một cuộc hội thoại.

ChatGPT sử dụng pretrain-model trên một lượng lớn dữ liệu văn bản (đến năm 2021) trước khi tinh chỉnh tham số phù hợp. Sau đó được huấn luyện bằng cách sử dụng học tăng cường và từ phản hồi của con người (RLHF).

ChatGPT ra đời được ứng dụng rộng rãi trong học tập, công việc: Tư tư vấn, chăm sóc sức khỏe, giáo dục, nghiên cứu. Nó giúp con người tối ưu về thời gian, nguồn lực.

Bên cạnh những ứng dụng cụ thể, ChatGPT vẫn còn nhiều hạn chế và thách thức như: dùng ChatGPT với mục đích xấu, trái đạo đức, khả năng đưa ra thông tin mới chính xác, cụ thể nhất còn hạn chế (vì chỉ được train đến năm 2021).

Trong tương lai, ChatGPT sẽ tiếp tục được nghiên cứu và phát triển để đáp ứng nhu cầu của xã hội.

## 2. Kiến trúc ChatGPT

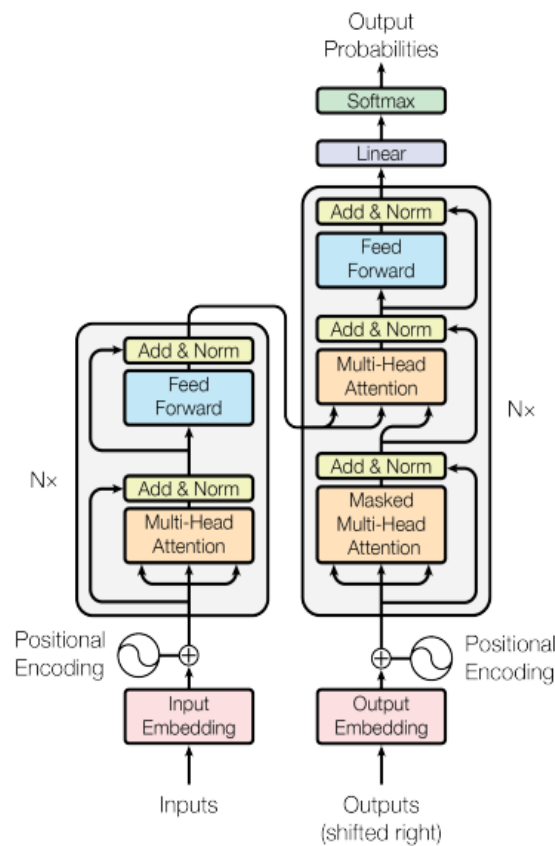
ChatGPT được thiết kế để tự động học và tạo ngôn ngữ tự nhiên bằng cách sử dụng các mạng neuron và kỹ thuật học sâu.

Kiến trúc của ChatGPT là hệ thống học máy có khả năng xử lý câu hỏi và tạo câu trả lời trong một đoạn hội thoại. Để làm được điều này ChatGPT sử dụng kiến trúc

Transformer, đây là một mạng nơ ron có thể xử lý dữ liệu chuỗi và hiểu ngữ cảnh của từ trong chuỗi đó. Transformer sử dụng lớp self-attention để xác định tầm quan trọng của từng từ trong chuỗi.

ChatGPT còn sử dụng các pretrain-model để đào tạo mô hình trên một lượng lớn dữ liệu văn bản trước khi tinh chỉnh để sử dụng cho các nhiệm vụ cụ thể, giúp mô hình học cách tự động và khách quan, từ đó nâng cao khả năng khái quát hóa của mô hình trong giải quyết các nhiệm vụ xử lý ngôn ngữ tự nhiên.

**\* Kiến trúc Transformer:**



Hình 1.1. Kiến trúc mạng Transformer

- Transformer được xây dựng trên cơ sở lớp self-attention kết hợp với các lớp mã hóa(encode) và giải mã(decode) để biểu diễn và xử lý các chuỗi đầu vào và đầu ra.
- **Ở bên trái là Encoder:** Gồm  $N_x = 6$  layers xếp chồng lên nhau, mỗi layers có multi-head self-attention và feed-forward, ngoài ra còn có các kết nối residual.
- **Ở bên phải là Decoder** cũng gồm  $N_x = 6$  layers xếp chồng lên nhau. Điểm khác biệt duy nhất so với lớp Encoder là có thêm lớp Masked Multi-head attention ở vị trí đầu tiên.
- **Attention:** Một attention function là việc ánh xạ query và một tập các cặp key-value đến đầu ra. Các query, key, value đều là các vector. Đầu ra (output) được tính bởi các trọng số của values, trong đó trọng số được tính với key tương ứng.
- + **Self-attention:** Các attention ở trong cùng một câu, từng thành phần trong câu tương tác với nhau. Mỗi token sẽ quan sát các tokens khác trong câu, cập nhật vector biểu diễn.
  - Query: Tìm kiếm thông tin xung quanh, hiểu ngữ cảnh giữa các tokens trong câu.
  - Key: Phản hồi yêu cầu của Query để tính trọng số Attention.
  - Value: Tính vector đại diện (attention vector).

Công thức tính attention vector:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

với  $d_k$  là số chiều của vector Key.

+ **Multi-head attention:** Mô hình có nhiều head hơn thay vì chỉ có 1 head như self-attention. Có khả năng tính toán song song attention function. Cho phép mô hình có thể lấy thông tin từ nhiều chiều không gian khác nhau.

Công thức:  $MultiHead(Q, K, V) = Concat(head_1, \dots, head_h)W$

với  $head_i = Attention(Q_i, K_i, V_i)$



- **Positional encoding**: Giúp cho mô hình biết được thứ tự của các token đầu vào (thể hiện vị trí của các từ trong câu).

- Cách hoạt động của mạng Transformer:

+ Input trước khi đưa vào Transformer sẽ cho qua lớp Word Embedding, input này sau đó đi qua Positional Encoding để xác định vị trí của câu input.

+ Sau đó phần input sẽ được đưa qua lớp Multi-Head Attention được tính toán một cách song song và trả ra output.

+ Output sẽ được normalize và đưa vào một mạng nơ ron rồi lại được normalize sau đó đưa ra ngoài nhờ vào kiến trúc Self Attention của transformer encoder. Qua từng lớp encoder, mô hình sẽ học được sự liên hệ giữa các từ trong câu input. Qua nhiều lớp Encoder, model sẽ học được ngữ nghĩa, context của câu input để đưa ra một feature vector đủ tốt dùng làm input cho việc phân loại.

**\* Kiến trúc GPT:**

- Triết lý mà openai tạo ra mô hình GPT: Biến đổi mô hình ngôn ngữ này đạt được hiệu quả cao trong việc xử lý nhiều nhiệm vụ khác nhau (Multitasks).

- **GPT-1**: 117 triệu tham số, với tập training data lấy từ 7000 cuốn sách khác nhau (khoảng 4.6GB text).

GPT-1 được xây dựng dựa trên mạng Transformer, nhưng bỏ phần encoder đi và giữ lại phần decoder, gồm 12 lớp decoder xếp chồng lên nhau. Mục đích làm nhiệm vụ cho việc dựa vào những từ phía trước để sinh ra những từ tiếp theo (transformer decoder). Mô hình chỉ học được theo một chiều từ trái sang phải, khác với BERT là học được cả hai chiều Transformer.

GPT-1 loại bỏ Multi-head Self-Attention giữa encoder và decoder.

+ Nhiệm vụ của GPT-1 là khám phá, cải tiến mạng Transformer Decoder, có thể thực hiện nhiều tác vụ khác nhau trong xử lý ngôn ngữ mà không cần sử dụng encoder (đầu vào). Cải thiện khả năng hiểu ngôn ngữ tự nhiên.

- **GPT-2**: Scale GPT-1 lên thành mô hình ngôn ngữ lớn với 1.5 tỷ tham số, hơn 40GB dữ liệu (Webtext).

Phát triển GPT-1 thành một mô hình ngôn ngữ sử dụng học không giám sát làm nhiều tác vụ khác nhau (không cần chuẩn bị tập data về supervised learning). Bắt đầu sử dụng prompt để mô hình học được từ các mẫu này.

+ Nhiệm vụ của GPT-2: Thử nghiệm trên tập dataset lớn và các phương pháp chuẩn bị dữ liệu cho việc training mô hình dựa trên nhiều tác vụ khác nhau.

- **GPT-3**: Tiếp tục scale GPT-2 lên thành 175 tỷ tham số, hơn 600 GB data. Lấy ra nhiều văn bản chất lượng cao, loại bỏ các văn bản chất lượng thấp.

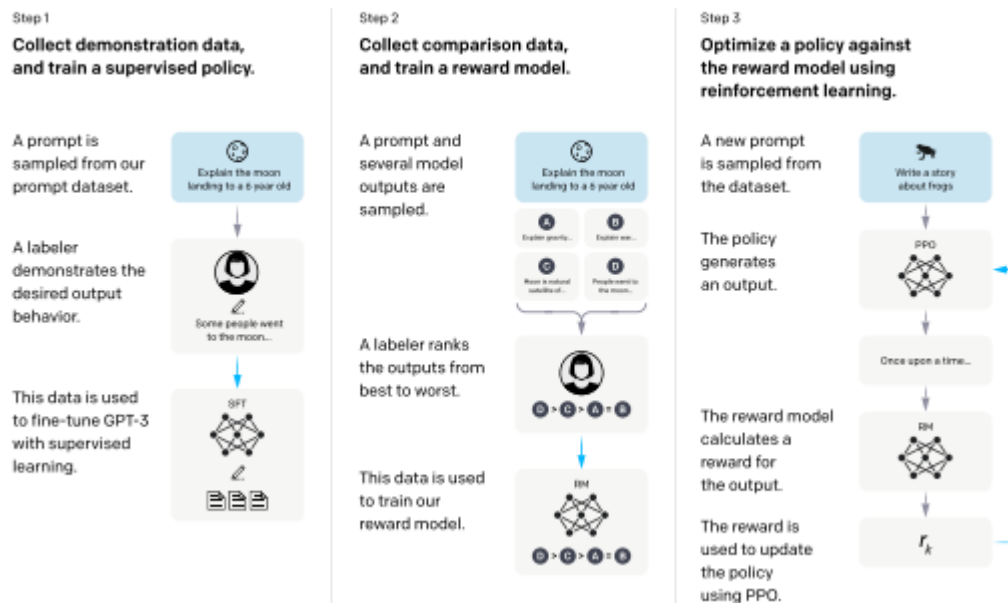
Tiếp tục đột phá mô hình GPT bằng cách sử dụng zero-shot/one-shot/few-shot learning mà không cần tinh chỉnh mô hình. Cung cấp các mẫu (prompt) nhiều hơn, chi tiết hơn so với GPT-2.

+ Nhiệm vụ của GPT-3:

- **GPT 3.5 (Instruction GPT)**: Bắt đầu đưa ra các prompt là những câu hỏi, fine-tuning thông qua các yêu cầu cụ thể.

Huấn luyện mô hình dựa theo phản hồi của con người. Kết hợp phản hồi liên tục của con người và học tăng cường để tránh những thông tin tiêu cực, những câu trả lời phi logic.

+ **Các bước:**



Hình 1.2. Cách hoạt động của ChatGPT

- **Bước 1: Supervised fine-tuning (SFT):** Thu thập dữ liệu và huấn luyện mô hình học có giám sát. Các nhãn được gán cung cấp hành vi mong muốn dựa trên prompt. Sau đó tinh chỉnh mô hình GPT-3 được huấn luyện trước bằng cách sử dụng phương pháp học có giám sát.
- **Bước 2: Reward model:** Thu thập dữ liệu để so sánh và huấn luyện trên phần thưởng (bước sau sẽ tốt hơn bước trước). Từ prompt đầu vào cho ra vài output khác nhau. Gán nhãn dữ liệu dựa trên thứ hạng cao nhất, tốt nhất phù hợp với người dùng. Sau đó huấn luyện mô hình để tìm ra output phù hợp nhất với con người.
- **Bước 3: Tối ưu hóa mô hình bằng cách sử dụng học tăng cường:** Sử dụng thuật toán PPO. Bước 2,3 được lặp lại liên tục, nhiều dữ liệu được so sánh để tìm ra output tốt nhất.

⇒ Mô hình GPT 3.5 phát triển phục vụ nhu cầu của con người. Là cơ sở để tạo nên ChatGPT thành công như hiện nay.

### 3. Đặc điểm của ChatGPT

#### 3.1 Lợi ích

**Khả năng hiểu ngôn ngữ tự nhiên:** ChatGPT được tạo ra để hiểu và xử lý ngôn ngữ tự nhiên của con người, giúp cho việc giao tiếp giữa máy tính và con người trở nên thuận tiện hơn. Điều này giúp người dùng tương tác với máy tính mà không cần các lệnh hoặc mã hoá phức tạp.

**Áp dụng trong Deep Learning:** ChatGPT được tạo ra bởi phương pháp học sâu, cho phép nó có thể học được các mẫu phức tạp và ngữ nghĩa trong dữ liệu văn bản. Như kiến trúc GPT-4, nó có thể học được từ hàng tỷ mẫu văn bản thông thường, cho phép nó nắm bắt hiệu quả các sử dụng từ vựng, ngữ pháp và ngữ nghĩa.

**Khả năng tổng hợp thông tin:** ChatGPT có khả năng tổng hợp và trích xuất thông tin từ dữ liệu lớn và từ nhiều nguồn, nó cung cấp câu trả lời chính xác và hữu ích cho người dùng. Điều này giúp người dùng tiết kiệm thời gian trong việc tìm kiếm thông tin và giải quyết vấn đề một cách nhanh chóng.

**Sáng tạo và linh hoạt:** ChatGPT có khả năng đưa ra các câu trả lời sáng tạo, linh hoạt, phù hợp với nhiều đối tượng, tình huống và yêu cầu của người sử dụng. Nó không chỉ cung cấp các câu trả lời phổ biến mà còn có thể đề xuất giải pháp mới cho một vấn đề mới phức tạp.

**Ứng dụng đa dạng trên nhiều lĩnh vực:** ChatGPT có thể được ứng dụng trong nhiều lĩnh vực và tình huống khác nhau như trợ lý ảo, hỗ trợ khách hàng, giáo dục, dịch thuật, viết lách, ... Điều này mở ra nhiều cơ hội cho các doanh nghiệp và cá nhân tận dụng công nghệ AI để nâng cao hiệu quả công việc.

**Hỗ trợ nhiều ngôn ngữ:** Mặc dù mức độ hỗ trợ có thể khác nhau, ChatGPT có khả năng xử lý trong nhiều ngôn ngữ khác nhau. Điều này giúp người dùng toàn cầu sử dụng mô hình này để giải quyết các vấn đề riêng.

### 3.2 Hạn chế

Mặc dù ưu điểm mà ChatGPT mang lại là rất nhiều, nhưng cần lưu ý rằng mô hình này chưa hoàn thiện. Đôi khi, nó có thể cung cấp thông tin không chính xác, lỗi thời hoặc không phù hợp. Hơn nữa, các mô hình có thể đưa ra phản hồi mơ hồ hoặc không liên quan đến nhu cầu của người dùng.

Một số hạn chế chính của ChatGPT:

+ **Hiểu biết hạn chế:** ChatGPT dựa trên dữ liệu huấn luyện từ internet và có kiến thức giới hạn từ nguồn dữ liệu đó. Chất lượng câu trả lời phụ thuộc vào chất lượng hướng dẫn, cụ thể trong câu hỏi (prompt). Việc trả ra một câu trả lời duy nhất như ChatGPT đang làm khiến cho thông tin hoàn toàn bị một chiều và không đa dạng và không cho chúng ta quyền lựa chọn và tìm hiểu. Nó không thể cung cấp thông tin mới và chính xác ngoài khả năng của dữ liệu huấn luyện ban đầu. Nếu dữ liệu huấn luyện không chứa thông tin đầy đủ hoặc chính xác về một chủ đề cụ thể, ChatGPT có thể không cung cấp câu trả lời chính xác.

+ **Thiếu kiểm soát về nội dung:** ChatGPT là một mô hình học máy dựa trên dữ liệu và không có khả năng phán đoán giá trị hoặc đúng sai của thông tin. Điều này có nghĩa là nó có thể tạo ra câu trả lời hoặc thông tin không chính xác, không phù hợp hoặc thiếu tính đạo đức. Việc kiểm soát nội dung và đảm bảo tính chính xác của thông tin là trách nhiệm của người sử dụng.

+ **Khả năng tạo ra thông tin gian lận:** ChatGPT có khả năng tạo ra thông tin gian lận hoặc đưa ra những lời khuyên không phù hợp. Điều này có thể xảy ra do sự tổng hợp thông tin từ dữ liệu huấn luyện hoặc do sự thiếu hiểu biết về các vấn đề phức tạp. Người

sử dụng cần có sự nhận thức và sàng lọc thông tin trước khi áp dụng vào quyết định hoặc hành động của mình.

+ ***Độ nhạy cảm với dữ liệu đầu vào***: ChatGPT có thể bị ảnh hưởng bởi từ ngữ hoặc câu truy vấn trong dữ liệu đầu vào. Một cách nói không chính xác hoặc không phù hợp có thể dẫn đến câu trả lời không đáng tin cậy hoặc không phù hợp. Sự chính xác và phù hợp của câu trả lời phụ thuộc một phần vào cách người sử dụng hỏi câu hỏi và sử dụng ngôn ngữ.

## Chương II: Bài toán phân tích sắc thái

**Phân tích sắc thái (SA)** là một trong những nhiệm vụ chính của lĩnh vực xử lý ngôn ngữ tự nhiên (NLP). SA là quá trình xác định và phân loại các cảm xúc khác nhau trong văn bản, ví dụ như cảm xúc tích cực, tiêu cực hoặc trung tính, hoặc các cảm xúc cụ thể như vui, buồn, tức giận hoặc ghê tởm, để đánh giá thái độ của con người đối với một chủ thể hoặc thực thể cụ thể, ...

Phân tích cảm xúc đóng vai trò quan trọng:

- + Con người bày tỏ suy nghĩ về cảm xúc của mình ngày càng nhiều hơn qua mạng xã hội, nên phân tích cảm xúc nhanh chóng trở thành một công cụ thiết yếu.

- + Tự động phân tích phản hồi của khách hàng, chẳng hạn như ý kiến trong phản hồi khảo sát trên mạng xã hội, cho phép thương hiệu tìm hiểu điều gì khiến khách hàng hài lòng hoặc thất vọng, để họ có thể điều chỉnh sản phẩm và dịch vụ nhằm đáp ứng nhu cầu của khách hàng.

Mục đích chính của phân tích sắc thái là giúp máy tính hiểu và đánh giá cảm xúc của con người trong văn bản. Điều này có thể hữu ích trong nhiều ứng dụng như: đánh giá phản hồi của khách hàng, theo dõi tâm trạng trên mạng xã hội, phân tích cảm xúc trong tin tức hoặc bài viết blog, và hỗ trợ trong việc tạo ra hệ thống gợi ý nội dung phù hợp với sắc thái của người dùng.

Tuy bài toán phân tích sắc thái đã có nhiều tiến bộ trong những năm gần đây, nhưng việc hiểu và đánh giá chính xác sắc thái trong ngôn ngữ vẫn là một thách thức đối với máy tính. Sự phức tạp của ngôn ngữ tự nhiên và tính tương đối của sắc thái khiến cho việc phân tích sắc thái trở thành một lĩnh vực nghiên cứu đa chiều và còn tiềm ẩn nhiều khía cạnh cần khám phá.

Phân tích sắc thái gồm nhiều cấp độ khác nhau như: phân tích sắc thái theo cấp độ tài liệu (SD), phân tích sắc thái theo cấp độ câu (SC), phân tích sắc thái dựa trên khía cạnh (ABSA).

**Khái niệm ý kiến:** Ý kiến của nhận xét, bình luận hay đánh giá của một người về một sản phẩm hoặc dịch vụ là những suy nghĩ, nhận định, tình cảm tâm lý hoặc cảm xúc của người đó về chất lượng, hình dáng, vật liệu hoặc giá cả của sản phẩm hoặc dịch vụ đó. Các ý kiến có thể là tích cực, tiêu cực hoặc trung lập tùy thuộc vào cảm nhận của từng cá nhân.

Thông thường, một đánh giá sẽ gồm các phần như sau:

- + Tiêu đề: mô tả ngắn gọn về đánh giá hoặc tên sản phẩm.
- + Loại sản phẩm: sản phẩm thuộc về loại nào, ví dụ như đầm hoặc túi xách.
- + Sản phẩm: sản phẩm sẽ được đánh giá.
- + Nội dung: ý kiến chi tiết của người dùng về sản phẩm.
- + Người đánh giá: người viết ý kiến đánh giá.
- + Mức đánh giá: thang điểm người đánh giá dành cho sản phẩm dựa trên mức độ hài lòng (1 sao, 2 sao, 3 sao, 4 sao hoặc 5 sao).
- + Thời gian đánh giá: ngày tháng năm mà ý kiến đánh giá được viết.

**Các loại ý kiến:** phân loại ý kiến thành 3 loại là ý kiến tích cực (positive), ý kiến tiêu cực (negative) và ý kiến trung lập (neutral).

+ **Ý kiến tích cực (positive)** là ý kiến chứa các thông tin tích cực về sản phẩm, những phản hồi đánh giá tốt, khen ngợi sản phẩm đó. Sản phẩm càng có nhiều ý kiến đánh giá tích cực thì người đọc sẽ càng cảm thấy sản phẩm đó tốt và chất lượng. Trong ý kiến tích cực thường xuất hiện các tính từ chỉ tính chất tích cực của sản phẩm như:



đẹp, xinh, dễ thương, tốt, tuyệt, ... và các từ chỉ cảm xúc tích cực của người viết như yêu, thích, hạnh phúc, vui, hài lòng, mong đợi, đề nghị...

+ **Ý kiến tiêu cực (negative)** là ý kiến chứa các thông tin tiêu cực về sản phẩm, những phản hồi đánh giá không tốt về sản phẩm đó. Sản phẩm có chất lượng thấp hoặc kiểu dáng chưa đẹp thường nhận được những ý kiến tiêu cực bởi vì khách hàng cảm thấy không hài lòng. Vì thế các nhà sản xuất sẽ xem xét, ghi nhận các ý kiến đánh giá để thay đổi, cải tiến các sản phẩm này cho tốt hơn. Trong ý kiến tiêu cực thường xuất hiện các tính từ chỉ tính chất tiêu cực của sản phẩm như chật, cứng, tệ, sai, xấu và các từ chỉ cảm xúc tiêu cực của người viết như ghét, thất vọng, ngó ngàng, trả lại, ...

+ **Ý kiến trung lập (neutral)** là ý kiến đánh giá chung chung về sản phẩm, có thể chứa cả nhận định tích cực và tiêu cực nhưng ở mức độ cân bằng. Ý kiến trung lập có thể chứa nội dung vừa khen vừa chê một sản phẩm, cũng có thể không khen không chê sản phẩm đó. Các ý kiến chứa các từ tạm được, ổn thường là các ý kiến trung lập. Ý kiến trung lập thì không phân cực.

Ứng dụng của bài toán phân tích sắc thái:

+ **Đánh giá sản phẩm và dịch vụ:** Phân tích sắc thái có thể được sử dụng để đánh giá sự hài lòng và cảm nhận của khách hàng đối với các sản phẩm và dịch vụ. Các doanh nghiệp có thể sử dụng phân tích sắc thái để hiểu ý kiến và phản hồi của khách hàng, từ đó cải thiện chất lượng sản phẩm và dịch vụ của mình.

+ **Phát hiện tin tức giả:** Phân tích sắc thái có thể giúp phát hiện và phân loại tin tức giả hoặc thông tin sai lệch trong các bài viết trên mạng. Các thuật toán phân tích sắc thái có thể phân tích các yếu tố như sự tin cậy, tính nhạy cảm, và sự khách quan của các bài viết để đưa ra đánh giá về tính chính xác của thông tin.

+ **Quản lý hỗ trợ khách hàng:** Phân tích sắc thái có thể được sử dụng để theo dõi và đánh giá sắc thái trong phản hồi của khách hàng đối với dịch vụ hỗ trợ. Điều này giúp

doanh nghiệp hiểu và đáp ứng nhanh chóng các vấn đề và yêu cầu của khách hàng, nâng cao chất lượng hỗ trợ khách hàng và tạo sự hài lòng.

+ ***Phân tích tâm trạng trong mạng xã hội***: Phân tích sắc thái có thể giúp đánh giá và theo dõi tâm trạng của người dùng trên mạng xã hội. Việc phân tích cảm xúc và sắc thái trong các bài đăng, bình luận và tin nhắn giúp hiểu được ý kiến, sự quan tâm và nhận định của người dùng về các chủ đề cụ thể hoặc sự kiện nổi bật.

+ ***Quảng cáo và tiếp thị***: Phân tích sắc thái có thể được sử dụng để đánh giá hiệu quả của các chiến dịch quảng cáo và tiếp thị. Điều này cho phép các nhà quảng cáo và nhà tiếp thị hiểu được phản ứng của khách hàng đối với quảng cáo và dự đoán hiệu quả của các chiến dịch quảng cáo tương lai.

+ Các ứng dụng khác bao gồm phân tích tâm lý, theo dõi thị trường và xác định ý kiến cộng đồng. Phân tích sắc thái cung cấp một phương pháp mạnh mẽ để hiểu và phân tích cảm xúc, ý kiến và tâm trạng trong văn bản, đóng góp vào việc đưa ra quyết định thông minh và cải thiện trải nghiệm người dùng.

## 1. Phân tích sắc thái cấp độ tài liệu (SD)

Phân tích sắc thái theo cấp độ tài liệu được coi là việc phân tích sắc thái theo cách truyền thống, chỉ tập trung vào việc phân loại sắc thái tổng thể của cả văn bản. Cấp độ văn bản được dùng để xác định mức độ cảm xúc của một đoạn văn (gồm hai hay nhiều câu văn) là tiêu cực, tích cực, hay trung lập.

Đây là một nhiệm vụ đầy thách thức với kích thước lớn của văn bản dẫn đến sự dư thừa từ ngữ và ý kiến, đôi khi trái ngược nhau trong cùng một tài liệu. Phân tích này đặc biệt hữu ích trong việc phân tích các bài báo và bài đăng trên blog về một vấn đề cụ thể sản phẩm hoặc công ty, và nó đòi hỏi sự tập trung cao độ, đặc biệt là khi chủ đề đang được thảo luận là nhạy cảm.

Phân tích sắc thái cấp độ tài liệu đòi hỏi hiểu và đánh giá không chỉ các câu hoặc từ trong tài liệu mà còn cả ngữ cảnh tổng thể và các mối quan hệ giữa các câu, đoạn văn. Bài toán này có thể được áp dụng cho các loại tài liệu khác nhau như bài viết, tin tức, đánh giá sản phẩm, văn bản pháp lý, văn bản y tế, và nhiều loại tài liệu văn bản khác.

SD được sử dụng nhiều khi ta muốn đánh giá sắc thái chung của một bài viết hay một chủ đề nào đó. Nhưng trong thực tế thì nó không đáp ứng được nhu cầu của doanh nghiệp khi cần xác định từng chi tiết nhỏ trong phản hồi của khách hàng.

**Ví dụ:** Bài viết ngắn cảm nhận về một cuốn sách:

*"Cuốn sách này thật tuyệt vời! Tôi đã đọc nhiều sách trong đời, nhưng đây là một trong những cuốn sách tuyệt nhất mà tôi từng đọc. Từ cách tác giả xây dựng câu chuyện đến cách ông miêu tả các nhân vật và hình ảnh đều vô cùng hấp dẫn và sống động.*

*Ngay từ những trang đầu tiên, cuốn sách đã thu hút sự chú ý của tôi và không buông được cho đến khi tôi đọc hết. Cốt truyện tuyệt vời và được xây dựng rất khéo léo, mang lại một sự kết hợp hoàn hảo giữa căng thẳng, hành động và cảm xúc. Tôi cảm thấy như mình đang sống trong thế giới mà tác giả đã tạo ra, và mỗi trang sách đều đưa tôi vào một cuộc phiêu lưu mới và thú vị.*

*Ngôn ngữ trong cuốn sách cũng rất tinh tế và đẹp mắt. Từ những câu chữ đơn giản cho đến mô tả tinh vi, tác giả đã tạo ra một dòng văn rất mượt mà và dễ đọc. Điều này giúp tôi hoàn toàn lắng nghe câu chuyện mà ông muốn kể và cảm nhận được sâu sắc những tình huống và tâm trạng của các nhân vật.*

*Cuối cùng, cuốn sách này để lại cho tôi một cảm giác sâu sắc và cảm động. Nó không chỉ là một câu chuyện giải trí mà còn chứa đựng những thông điệp và ý nghĩa sâu sắc về cuộc sống và con người. Tôi tin chắc rằng cuốn sách này là một trải nghiệm đáng nhớ đối với bất kỳ ai yêu thích văn học và tìm kiếm những cuốn sách đầy cảm hứng."*

⇒ Bài viết trên là một bài viết tích cực về một sản phẩm sách.

## 2. Phân tích sắc thái cấp độ câu (SC)

Phân tích sắc thái theo cấp độ câu làm nhiệm vụ xác định sắc thái (tích cực, tiêu cực hay trung lập) của mỗi câu trong một đoạn văn bản dựa trên nội dung và ngữ cảnh của câu văn đó.

Thay vì chỉ phân tích sắc thái tổng thể của cả văn bản, SC tập trung vào mức độ phân tích chi tiết hơn, tách riêng từng câu để xác định ý kiến, quan điểm cụ thể trong từng câu.

Phân tích sắc thái cấp độ câu yêu cầu máy tính hiểu và đánh giá các cảm xúc, ý kiến hoặc tâm trạng được diễn đạt trong một câu. Điều này có thể hữu ích trong nhiều ứng dụng, bao gồm đánh giá phản hồi của khách hàng, phân tích cảm xúc trong mạng xã hội, phát hiện tin tức giả và tạo ra hệ thống tự động phản hồi.

SC được sử dụng nhiều khi ta cần phân tích một mô tả ngắn (từ bình luận ngắn trên mạng xã hội, đánh giá ngắn gọn về sản phẩm, dịch vụ của khách hàng).

**Ví dụ:** Bình luận về một bộ phim mới ra mắt:

+ **Câu 1:** "*Diễn xuất của các diễn viên trong phim rất xuất sắc, đặc biệt là vai diễn chính.*" ⇒ Tích cực.

+ **Câu 2:** "*Kịch bản của phim thiếu logic và không thực tế.*" ⇒ Tiêu cực.

+ **Câu 3:** "*Âm nhạc và hình ảnh trong phim tuyệt đẹp, tạo nên một trải nghiệm đầy màu sắc.*" ⇒ Tích cực.

Phát biểu bài toán phân tích sắc thái cấp độ câu trong nghiệp vụ mua sách trên trang thương mại điện tử:

+ Đầu vào: Các bình luận của người mua về sách như dịch vụ, sản phẩm,...

+ Đầu ra: Cảm xúc tương ứng được đề cập trong câu bình luận của người mua (tích cực, tiêu cực, trung lập).

### 3. Phân tích sắc thái theo khía cạnh (ABSA)

Phân tích sắc thái theo khía cạnh (Aspect-Based Sentiment Analysis) làm nhiệm vụ xác định các cặp (khía cạnh - sắc thái) của một câu trong văn bản. Tùy vào mục đích mà chúng ta có thể chỉ lựa chọn ra những khía cạnh đặc trưng nhất để đưa ra các phân tích, đánh giá.

Phân tích sắc thái theo khía cạnh yêu cầu máy tính hiểu và đánh giá sắc thái của một văn bản từng phần theo các khía cạnh như giá trị sản phẩm, dịch vụ, tính năng, đáng tin cậy, hỗ trợ khách hàng, và nhiều khía cạnh khác. Điều này có thể hữu ích trong việc đánh giá và cải thiện các khía cạnh cụ thể của một sản phẩm, dịch vụ hoặc trải nghiệm người dùng.

Các nhiệm vụ của phân tích sắc thái theo khía cạnh:

+ ***Aspect Category Classification***: Xác định cặp chủ đề và khía cạnh mà một ý kiến được thể hiện trong văn bản. Chủ đề và khía cạnh nên được chọn từ một tập hợp các loại chủ đề đã được xác định (ví dụ: laptop, nhà hàng, thực phẩm) và theo khía cạnh (ví dụ: giá cả, chất lượng).

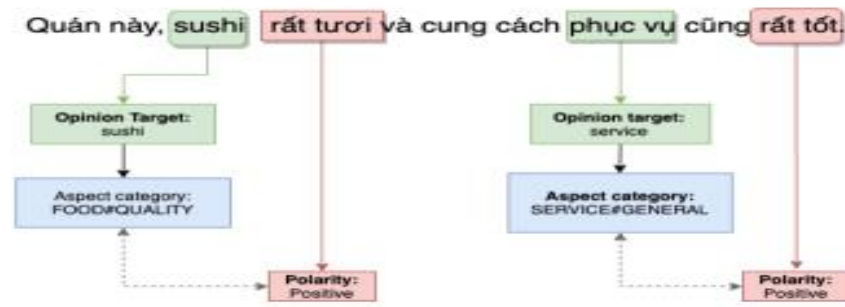
+ ***Opinion Target Expression***: Trích xuất ngôn ngữ được sử dụng trong văn bản đầu vào đề cập đến thực thể được nói tới cho mỗi cặp khía cạnh thực thể.

+ ***Aspect Sentiment Classification***: Dự đoán phân loại cảm xúc cho một khía cạnh cụ thể trong câu.

**Ví dụ:** Hình dưới đây mô tả ba nhiệm vụ của ABSA:

+ Trích xuất mục tiêu ý kiến: “sushi” và “phục vụ”.

- + Xác định thực thể-khía cạnh: “Food-Quality” và “Service-General”.
- + Xác định cảm xúc của một khía cạnh: “Positive” và “Negative”.



Hình 1.3.Nhiệm vụ của ABSA

Ví dụ khi mua sách trên trang thương mại điện tử, các khía cạnh như chất lượng, giá cả, nội dung, dịch vụ,... được nói tới.

ABSA giúp cho doanh nghiệp đánh giá chi tiết hơn về phản hồi của khách hàng đối với sản phẩm, dịch vụ, từ đó có thể cải thiện chất lượng, tăng cường trải nghiệm khách hàng,...

Phát biểu bài toán phân tích sắc thái theo khía cạnh trong nghiệp vụ mua sách trên trang thương mại điện tử:

- + Đầu vào: Các bình luận của người mua về sách như dịch vụ, sản phẩm,...
- + Đầu ra: Đưa ra cặp khía cạnh-sắc thái cho từng câu (ví dụ: chất lượng-tích cực, dịch vụ-tiêu cực, ...)

## Chương III: ChatGPT và việc đánh nhãn dữ liệu

### 1. Khảo sát khả năng đánh nhãn dữ liệu của ChatGPT qua các test case

Phân loại một câu bình luận về chủ đề mua sách trên trang thương mại điện tử xem sắc thái của câu đó mang ý nghĩa tích cực, tiêu cực hay trung lập. Thực hiện hỏi ChatGPT thông qua giao diện hội thoại mà openai tạo ra.

\* **Câu bình luận chỉ mang một sắc thái duy nhất**

- **Tích cực:**

+ **Câu hỏi 1:** Hãy cho tôi biết câu bình luận sau đây là tích cực, tiêu cực hay trung lập:  
*“Sách xoay quanh những câu chuyện thường ngày hay những trend của giới trẻ hiện nay. Có hình ảnh, câu nói hài hước vui vẻ. Ngôn ngữ bình dị, dễ gần từ đó dễ lấy được lòng yêu thích của các bạn trẻ hiện nay. Nhân viên giao hàng vui vẻ 🥰. các bạn trẻ nên đọc qua 1 lần, bạn sẽ cười khi đến hết cuốn sách mà thôi.”*

⇒ **ChatGPT trả lời:** Câu bình luận này là tích cực.

+ **Câu hỏi 2:** Hãy cho tôi biết câu bình luận sau đây là tích cực, tiêu cực hay trung lập:  
*“Một cuốn sách gợi nhớ tuổi thơ của bao thế hệ 8x và 9x. Cảm ơn hai chị tác giả đã cho ra mắt quyển sách hay thế này. Tiki giao hàng sớm. Có lớp bọc bên ngoài nên sách trông rất mới.”*

⇒ **ChatGPT trả lời:** Câu bình luận này là tích cực.

+ **Câu hỏi 3:** Hãy cho tôi biết câu bình luận sau đây là tích cực, tiêu cực hay trung lập:  
*“Tuổi thơ lớn lên cùng những cuốn truyện của Nguyễn Nhật Ánh. Đúng là sách của Nguyễn Nhật Ánh không chỉ dành cho trẻ con, mà còn dành cho những ai từng là trẻ*

con. Cực kỳ hài lòng về Tiki, giao hàng siêu nhanh, mình đặt mua lúc 23h30, trưa ngày hôm sau đã giao, đóng gói bảo đảm, chắc chắn.”

⇒ **ChatGPT trả lời:** Câu bình luận này là tích cực.

- **Tiêu cực:**

+ **Câu hỏi 1:** Hãy cho tôi biết câu bình luận sau đây là tích cực, tiêu cực hay trung lập: “Tôi không hài lòng với khâu vận chuyển.”

⇒ **ChatGPT trả lời:** Câu bình luận trên là tiêu cực.

+ **Câu hỏi 2:** Hãy cho tôi biết câu bình luận sau đây là tích cực, tiêu cực hay trung lập: “Tuy có bọc sách, nhưng mở ra nhìn cứ như đã có người lật 1 2 trang đầu xem rồi, trong sách để hàng thật 236 trang nội dung chất lượng in sách rõ nét, mà cuốn này chỉ có 232 trang, bìa sách bị mờ nữa, với một đĩa yêu sách như mình thật sự rất thất vọng.”

⇒ **ChatGPT trả lời:** Câu bình luận này là tiêu cực.

+ **Câu hỏi 3:** Hãy cho tôi biết câu bình luận sau đây là tích cực, tiêu cực hay trung lập: “sách bị dơ, bìa sách bị cong, không có bao sách. gói hàng sơ sài, không có chống sốc. lại còn trời mưa nữa chứ, may mà không ướt, ướt chắc bỏ sách hết...nói chung lần đầu mua mà mắc quạo!!! 🙄🙄🙄🙄”

⇒ **ChatGPT trả lời:** Câu bình luận này là tiêu cực.

**Nhận xét:** Đối với những câu chỉ mang một sắc thái duy nhất, ChatGPT sẽ đánh nhãn là một trong hai sắc thái là tích cực hoặc tiêu cực.

\* **Câu văn mang nhiều sắc thái khác nhau**

+ **Câu hỏi 1:** Hãy cho tôi biết câu bình luận sau đây là tích cực, tiêu cực hay trung lập: “Sách giao hơi chậm, đóng gói không chắc chắn mình không biết do quá trình vận chuyển hay sách đã bị quăn nếp gấp ở 2 đầu như vậy trước. Về sản phẩm mình khá thích



tô màu và vẽ, sách này giúp mình cảm thấy như được tìm về niềm đam mê hồi ấy :)) Cứ mỗi trang mở ra là 1 trang tô màu và 1 trích dẫn nhỏ.”

⇒ **ChatGPT trả lời:** Câu bình luận trên có thể được xem là trung lập

+ **Câu hỏi 2:** Hãy cho tôi biết câu bình luận sau đây là tích cực, tiêu cực hay trung lập:  
“tiki giao hàng nhanh hơn dự kiến nhưng đóng gói còn lỏng lẻo, không chắc chắn, có 2 mẫu giấy to trong hộp, không biết để làm gì, hi vọng tiki sẽ dùng bao chống sốc để hạn chế hư hại. mình mua 8 quyển, đây là quyển duy nhất có seal, nên không bị vấn đề gì. hi vọng là tiki sẽ khắc phục lại phần đóng gói.”

⇒ **ChatGPT trả lời:** Câu bình luận trên có thể được xem là trung lập.

+ **Câu hỏi 3:** Hãy cho tôi biết câu bình luận sau đây là tích cực, tiêu cực hay trung lập:  
“Thứ nhất: Về chất lượng sách thì mình rất hài lòng, nội dung sách rất hay và ý nghĩa, Tiki giao hàng khá nhanh Thứ hai: sách Tiki giao không có bookmark mặc dù mình đặt bản thông thường có kèm theo bookmark, bookmark của sách rất đẹp và mình là người khá nghiện nó nên mong Tiki sẽ khắc phục tình trạng này trong những lần tới. Bởi vậy nên mình đánh 4 sao thôi mặc dù sách đáng được 5 sao.”

⇒ **ChatGPT trả lời:** Câu bình luận trên có thể được xem là trung lập.

**Nhận xét:** Dựa theo rất nhiều test case khảo sát thì ChatGPT chỉ có thể đưa ra nhãn là trung lập khi câu văn mang 2 sắc thái tích cực và tiêu cực trong đó (dù sắc thái tích cực hay tiêu cực nhiều hơn bên còn lại). Để khắc phục điều này thì có một hướng đi khác cho bài toán đó là ta sẽ đánh nhãn theo khía cạnh.

\* **Đánh nhãn theo khía cạnh.**

- **Câu hỏi:** Đây là câu về chủ đề bình luận khi mua sách trên trang thương mại điện tử.

Hãy thực hiện các yêu cầu sau:

1- Xác định các khía cạnh, đối tượng cho câu văn bằng cách xác định các từ khóa liên quan về chủ đề sách (nội dung, chất lượng, đóng gói, giao hàng, dịch vụ, giá cả), đưa ra tối đa 3 khía cạnh, nếu không có khía cạnh nhắc tới thì ghi là None.

2- Đưa ra một trong ba sắc thái: positive, negative, neutral cho từng khía cạnh đó.

3- Cột "Decisions" bao gồm khía cạnh và sắc thái với cú pháp: {"khía cạnh": "sắc thái"}

**Câu 1:** Hình thức đẹp, còn nguyên màng co. Nội dung thì cũng là văn phong của Nguyễn Nhật Ánh trước giờ chứ không mới mẻ đâu. Là mình thì thấy không hay như truyện Quán gò đi lên hay Nữ sinh. Mình sưu tầm và đọc tất cả sách Nguyễn Nhật Ánh nên đáng mua lắm. Đặc biệt có vài câu thơ khá hay. dẫu trang đẹp.

**Câu 2:** Sách giao hơi chậm, đóng gói không chắc chắn mình không biết do quá trình vận chuyển hay sách đã bị quăn nếp gấp ở 2 đầu như vậy trước. Về sản phẩm mình khá thích tô màu và vẽ, sách này giúp mình cảm thấy như được tìm về với niềm đam mê hồi ấy :)) Cứ mỗi trang mở ra là 1 trang tô màu và 1 trích dẫn nhỏ.

**Câu 3:** Tiki giao hàng nhanh lắm, nhân viên giao hàng cũng nhiệt tình nữa! Sách thì xinh khỏi bàn luôn, giấy dày, nhưng sách có chỗ hơi bần một tí ạ. Sách tô rất dễ với hình nhìn yêu nữa... Một buổi chiều ngồi nhâm nhi và tô màu thì tuyệt zời luôn ấy!!! Mình mua đúng lúc bán ấy nên rẻ kinh khủng. nói chung là nên mua nha cả nhà!!!!

⇒ ChatGPT trả lời:

Câu	Nội dung	Chất lượng	Đóng gói	Giao hàng	Dịch vụ	Giá cả
1	NEG	POS	POS	NAN	NAN	NAN
2	NAN	POS	NEG	NEG	NAN	NAN
3	NAN	POS	NAN	POS	NAN	POS

Bảng 1. Đánh nhãn theo khía cạnh

⇒ Như vậy từng khía cạnh cụ thể sẽ mang một sắc thái nhất định, giảm bớt sự nhập nhằng về nghĩa với những câu văn mang nhiều sắc thái.

## 2. Chiến lược

- Sử dụng API do OpenAI cung cấp: Chọn mô hình text-davinci-003, sử dụng thông qua API key có sẵn của ChatGPT để có thể tích hợp vào chương trình Python, làm tác vụ gán nhãn dữ liệu.

+ Ưu điểm: Sử dụng API của OpenAI, đảm bảo khả năng đánh nhãn dữ liệu được chính xác và khách quan.

+ Nhược điểm: Cần phải trả phí khi sử dụng API (OpenAI chỉ cung cấp 18 \$ miễn phí cho người dùng).

⇒ Cách khắc phục: Kết hợp gán nhãn thủ công do hiện nay chưa có đơn vị nào tài trợ về vấn đề này.

- Sử dụng các trình tự động điều khiển tự động trang web như selenium để tương tác với chatgpt qua web: <https://chat.openai.com>.

+ Ưu điểm: Tự động hóa quá trình hỏi chatgpt qua web, giúp tiết kiệm được thời gian, công sức.

+ Nhược điểm:

- Các phần tử trang web bị thay đổi thuộc tính liên tục (tên class,...).
- Nếu gọi quá nhiều câu hỏi trong một khoảng thời gian ngắn thì ChatGPT không đưa ra câu trả lời mà bắt chờ đợi ⇒ Cần kết hợp gán nhãn thủ công khi đó.

## 3. Quy trình

### 3.1. Quy trình đánh nhãn dữ liệu qua API

- **Bước 1:** Lấy API key mà OpenAI cung cấp bằng cách vào trang web: <https://www.openai.com> rồi đăng nhập vào tài khoản của mình, vào phần API key để lấy API có sẵn hoặc có thể tự tạo cho riêng mình.

Ví dụ về một API key:

```
sk-shqMw3Os7TOFEEdTEVvk80T3BlbkFJIE8RsKIVEFUg5vBsdSuQ
```

- **Bước 2:** Thiết lập API key: Sao chép mã API key vào một file text (để bảo mật) rồi gán vào một biến `openai.api_key` để sử dụng lại.

```
with open("B:\\Đồ án I\\Project I\\ChatGPT labelling\\apikey.txt", "r") as f:  
    openai.api_key = f.readline()
```

- **Bước 3:** Gửi yêu cầu cho một mô hình của ChatGPT (text-davinci).

```
model = 'text-davinci-003'  
  
def get_response_from_chatgpt(user_question):  
    response = openai.Completion.create(  
        engine= model, prompt = user_question,  
        max_tokens = 128, n = 1, temperature = 0.5)  
    response_text = response.choices[0].text  
    return response_text
```

- **Bước 4:** Tạo câu lệnh (Prompt) phù hợp với bài toán để chuẩn bị đưa vào ChatGPT.
- **Bước 5:** ChatGPT đưa ra câu trả lời và sử dụng câu trả lời đó để đánh nhãn dữ liệu.

### 3.2 Quy trình đánh nhãn dữ liệu bằng selenium

- **Bước 1:** Đọc prompt được viết sẵn từ một file trước đó.
- **Bước 2:** Đọc 5 câu một từ file gồm nhiều câu (khoảng 10000 câu), mỗi câu sẽ được đánh 1 nhãn riêng.

- **Bước 3:** Truy cập tự động trang web <https://chat.openai.com/auth/login> rồi tương tác các phần tử trang web chatgpt bằng selenium.

- **Bước 4:** Lấy câu trả lời rồi viết ra file text.

## 4. Kỹ thuật

### \* Đánh nhãn dữ liệu nhanh.

Mỗi lần gọi model của ChatGPT có thể đánh được trung bình từ 5-10 câu bình luận tùy thuộc vào số lượng từ (token) có thể được gọi.

Dùng model text-davinci có sẵn của ChatGPT nên nó có thể requests tối đa 4096 tokens cho mỗi lần gọi.

- Trường hợp thỏa mãn  $< 4096$  tokens:

### + Đánh nhãn theo cấp độ câu.

**Câu hỏi:** Hãy cho tôi biết các câu bình luận sau đây là tích cực, tiêu cực hay trung lập:  
51, "Sách nhỏ đẹp, nội dung hơi kiệm chữ nhưng vẫn cuốn hút người đọc. Hơi buồn vì tiền ship bằng tiền sách." 52, "Ship hàng siêu nhanh ạ, anh giao hàng cũng thân thiện. Sách pé pé xinh xinh cứng lắm, nội dung thú vị độc đáo, nói chung là hồng có j để chê ạ" 53, "Sách cứng xui nà! đang stress mà nhận được sách đọc vài trang mà quên hết mọi sự, cười rớt hàm." 54, "Sách nhỏ gọn Giao hàng đúng hẹn Đọc đi đọc lại cuốn sách vẫn thấy sự tấu hài sâu sắc và đúng với thực tế cuộc sống Sau vui vẻ không quạu là có từ điển tiếng ""em"" 55, "Sách nhẹ nhàng, dễ thương. Đọc không cần chuẩn bị tâm lý, kiến thức gì, cũng không cần suy nghĩ gì quá nhiều. Đúng với định nghĩa từ điển" 56, sách rất hay. nhưng ngắn quá. muốn nhiều hơn nữa 💜 57, OK 58, "Sách dễ thương, tiki giao hàng nhanh sáng đặt trưa đã giao. ❤️❤️❤️" 59, Nếu nhận sách mà bìu sách sạch sẽ thì tuyệt vời biết bao 🤩 60, "Trưa nay được tận tay sờ nắm cả 2 bản cứng và mềm mà sung sướng cười cả ngày luôn ấy. ^^ Sách được in ấn tỉ mỉ và cẩn thận. Rất hài lòng vì chất

lượng sách. Chỉ tham lam thêm, nếu postcard của bản bì cứng cũng có các đoạn thơ dài như bản mềm thì siêu mãn ý luôn (vì bản mềm là của chị khác mua cùng rồi, chẳng lẽ mua thêm quyển =D )."

⇒ **ChatGPT trả lời:** 51- Trung lập, 52- Tích cực, 53- Tích cực, 54- Tích cực, 55- Tích cực, 56- Tiêu cực, 57- Trung lập, 58- Tích cực, 59-Tiêu cực, 60-Trung lập.

+ **Đánh nhãn theo khía cạnh.**

**Câu hỏi:** Đây là các câu về chủ đề bình luận khi mua sách trên trang thương mại điện tử. Hãy thực hiện các yêu cầu sau: 1- Xác định các khía cạnh, đối tượng cho từng câu bằng cách xác định các từ khóa liên quan về chủ đề sách (nội dung, chất lượng, đóng gói, giao hàng, dịch vụ, giá cả), đưa ra tối đa 3 khía cạnh 2- Đưa ra một trong ba sắc thái: positive, negative, neutral cho từng khía cạnh đó. Các câu: 85749, "dịch vụ của tiki vô cùng tốt. giao hàng rất nhanh. mình đặt rạng sáng hôm nay thì chiều đã nhận được boxset. tuy nhiên boxset của mình lại rách 1 vết. buồn không tả được. mua truyện sưu tầm mà thế này buồn cả ngày ấy. thôi thì sách bên trong đẹp, chất lượng dịch vụ tốt. mỗi tội sp mình bị lỗi do nhân phẩm thôi :((" 85754, "thích anime, mua manga để sưu tập thêm. khá bất ngờ vì nét vẽ của manga lại tuyệt vời như thế! nét vẽ đẹp cực kỳ, nội dung độc đáo. cá nhân mình không thích loại boxset của kim đồng lâm, chất liệu giống hộp carton gấp lại. trong tập 1 còn lỗi sai chính tả, dịch thuật không ổn lắm sang các tập sau thì hết rồi. dịch vụ của tiki tốt: bảo quản, đóng gói cẩn thận. giao hàng đúng thời gian." 85822, "cầm nang dày và chi tiết. thái lan như thu nhỏ trong lòng bàn tay. tiki giao hàng nhanh, dịch vụ tốt!" 86427, "quyển sách đầy đủ và chi tiết đi từ những kiến thức tổng quát căn bản đến những kiến thức chuyên ngành kiểm toán được trình bày khoa học, hợp lý. tiki giao hàng nhanh, dịch vụ tốt!" 86691, "giá sale 12k cho một cuốn sách mới tinh về hình thức, chất lượng về nội dung. dịch vụ tiki chuẩn chỉ nữa. 10 điểm" 86776, đúng hàng và mới nguyên rất hài lòng về sản phẩm và dịch vụ của tiki. **Khía cạnh:** Nội dung, Chất lượng, Đóng gói, Giao hàng, Dịch vụ, Giá cả.

⇒ **ChatGPT trả lời:** 85749: Nội dung: Neutral, Chất lượng: Neutral, Đóng gói: Negative, Giao hàng: Positive, Dịch vụ: Positive, Giá cả: Neutral.

85754: Nội dung: Positive, Chất lượng: Positive, Đóng gói: Negative, Giao hàng: Positive, Dịch vụ: Positive, Giá cả: Positive.

85822: Nội dung: Positive, Chất lượng: Positive, Đóng gói: Positive, Giao hàng: Positive, Dịch vụ: Positive, Giá cả: Neutral.

86427: Nội dung: Positive, Chất lượng: Positive, Đóng gói: Positive, Giao hàng: Positive, Dịch vụ: Positive, Giá cả: Neutral.

86691: Nội dung: Positive, Chất lượng: Positive, Đóng gói: Positive, Giao hàng: Positive, Dịch vụ: Positive, Giá cả: Positive.

86776: Nội dung: Positive, Chất lượng: Positive, Đóng gói: Positive, Giao hàng: Positive, Dịch vụ: Positive, Giá cả: Positive.

- Trường hợp > 4096 tokens sẽ báo lỗi:

**InvalidRequestError:** This model's maximum context length is 4097 tokens, however you requested 4099 tokens (3075 in your prompt, 1024 for the completion) Please reduce your prompt, or completion length.

\* **Đánh nhãn dữ liệu tự động sử dụng API key.**

Sử dụng vòng lặp để liên tục gọi đến ChatGPT đánh nhãn dữ liệu cho từng batch nhỏ (5-10 câu một lần gọi). Viết các câu trả lời của ChatGPT ra một file text.

- **Đánh nhãn theo cấp độ câu.**

question = 'Hãy cho tôi biết các câu bình luận sau đây là tích cực, tiêu cực hay trung lập. Ứng với mỗi câu bình luận thì đưa ra câu trả lời là số thứ tự câu + một trong các từ sau: positive, negative, neutral '

```
def label_sentiment(comment):  
    response = openai.Completion.create(  
        engine = model, prompt = question + comment,  
        max_tokens = 48, n = 1, temperature = 0.5)  
    return response.choices[0].text  
with open('B:\\Đồ án I\\Project I\\ChatGPT labelling\\label.txt', 'w', encoding =  
'utf-8') as file:  
    for i in range(0, 10, 5):  
        sentiment=label_sentiment(comments[i] + comments[i+1] + comments[i+2]  
+ comments[i+3] + comments[i+4])  
        labeled_data.append(sentiment)  
        file.write(sentiment)  
print(labeled_data)
```

**Kết quả:** Với batch\_size là 5 câu thì 5 câu một được tự động gán nhãn:

Câu 0: positive  
Câu 1: positive  
Câu 2: positive  
Câu 3: positive  
Câu 4: neutral  
  
Câu 5: Neutral  
Câu 6: Positive  
Câu 7: Positive  
Câu 8: Positive  
Câu 9: Positive

- **Đánh nhãn theo khía cạnh.**



```
question = 'Xác định các khía cạnh, đối tượng cho từng câu bằng cách xác định  
các từ khóa liên quan về chủ đề sách (nội dung, chất lượng, đóng gói, giao hàng,  
dịch vụ, giá cả), đưa ra tối đa 3 khía cạnh; Đưa ra một trong ba sắc thái: positive,  
negative, neutral cho từng khía cạnh đó'  
  
def label_sentiment(comment):  
    response = openai.Completion.create(engine = model,  
        prompt = question + comment, max_tokens = 1024,  
        n = 1, temperature = 0.5)  
    return response.choices[0].text  
  
with open('B:\\Đồ án I\\Project I\\ChatGPT labelling\\aspect\\label_aspect.txt',  
    'w', encoding = 'utf-8') as file:  
    for i in range(0, 10, 5):  
        sentiment=label_sentiment(comments[i] + comments[i+1] + comments[i+2]  
+ comments[i+3] + comments[i+4])  
        labeled_data.append(sentiment)  
        file.write(sentiment)  
    print(labeled_data)
```

**- Kết quả:**

Câu 1:

Khía cạnh 1: Nội dung - Neutral

Khía cạnh 2: Chất lượng - Positive

Khía cạnh 3: Đóng gói, giao hàng, dịch vụ, giá cả - Positive

Câu 2:

Khía cạnh 1: Nội dung, chất lượng: Positive

Khía cạnh 2: Đóng gói, giao hàng: Positive

Khía cạnh 3: Giá cả: Positive

**\* Đánh nhãn dữ liệu tự động sử dụng selenium.**

**- Đánh nhãn theo cấp độ câu.**

+ Prompt:

Hãy cho tôi biết các câu bình luận sau đây là tích cực, tiêu cực hay trung lập (Đưa ra câu trả lời với định dạng sau: STT - Sắc thái).

+ Đọc prompt từ file prompt.txt.

```
with open('B:\\Đồ án I\\Project I\\selenium automatic chatgpt\\prompt.txt', 'r',  
encoding = 'utf-8') as file:  
    prompt = file.read()
```

+ Đọc các câu bình luận từ file comment.txt.

```
with open('B:\\Đồ án I\\Project I\\selenium automatic chatgpt\\comment.txt', 'r',  
encoding='utf-8') as file:  
    comments = []  
    for line in file:  
        comments.append(line)  
    summarize = []  
    for i in range(0, len(comments)-5, 5):  
        summarize.append(comments[i]+comments[i+1]+comments[i+2]+  
comments[i+3] + comments[i+4])
```

+ Điều khiển trình duyệt tự động bằng selenium.

```
# Set up chrome driver  
chrome_options = Options()  
driver = uc.Chrome(chrome_options=chrome_options)  
  
# Mở trang web ChatGPT
```

```
driver.get('https://chat.openai.com/auth/login')
# Đăng nhập tài khoản chatgpt
button_login = driver.find_element(By.XPATH, "//button[@class='btn relative btn-primary']")[1]")
button_login.click()
time.sleep(3)
account = 'quangminhpulisic123@gmail.com'
password = 'minhboy123'
# Nhập tài khoản, mật khẩu
mail = driver.find_elements(By.TAG_NAME, "input")[1]
mail.send_keys(account)
btn = driver.find_elements(By.TAG_NAME, "button")[0]
btn.click()
pw = driver.find_elements(By.TAG_NAME, "input")[2]
pw.send_keys(password)
btn = driver.find_element(By.XPATH, "//button[@class= 'c89f1057d c08709d93 cfd7e7ce c948a708e _button-login-password']")
btn.click()
time.sleep(2)
btn1 = driver.find_element(By.XPATH, "//button[@class = 'btn relative btn-neutral ml-auto']")
btn1.click()
time.sleep(2)
btn2 = driver.find_element(By.XPATH, "//button[@class = 'btn relative btn-neutral ml-auto']")
btn2.click()
time.sleep(2)
```

```
btn3 = driver.find_element(By.XPATH, "//button[@class = 'btn relative btn-
primary ml-auto']")
btn3.click()
time.sleep(2)
# Dùng vòng lặp để viết câu trả lời ra file text
for i in range(3):
    text_area = driver.find_element(By.TAG_NAME, "textarea")
    driver.execute_script("arguments[0].value = arguments[1];", text_area, prompt
+ summarize[i])
    text_area.send_keys(Keys.ENTER)
    btn_answer = driver.find_element(By.XPATH, "//button[@class = 'absolute p-
1 rounded-md md:bottom-3 md:p-2 md:right-3 dark:hover:bg-gray-900
dark:disabled:hover:bg-transparent right-2 disabled:text-gray-400 enabled:bg-
brand-purple text-white bottom-1.5 transition-colors disabled:opacity-40']")
    btn_answer.click()
    time.sleep(20)
    answer = driver.find_elements(By.XPATH, "//div[@class = 'markdown prose
w-full break-words dark:prose-invert light']")[i]
    result = answer.text
    print(result)
with open('B:\\Đồ án I\\Project I\\selenium automatic chatgpt\\result.txt', 'a',
encoding = 'utf-8') as file:
    file.write(result + '\n')
    time.sleep(5)
driver.quit()
```

+ Kết quả được viết ra file result.txt:

Các câu bình luận đã cho có các sắc thái như sau:

0 - Trung lập

1 - Tích cực

2 - Tích cực

3 - Tích cực

4 - Trung lập

Các câu bình luận đã cho có các sắc thái như sau:

5 - Trung lập

6 - Tích cực

7 - Tích cực

8 - Tích cực

9 - Tích cực

Các câu bình luận đã cho có các sắc thái như sau:

10 - Trung lập

11 - Trung lập

12 - Tiêu cực

13 - Tích cực

14 - Tích cực

- **Đánh nhãn theo khía cạnh.**

+ Prompt:

Đây là các câu về chủ đề bình luận khi mua sách trên trang thương mại điện tử

Hãy thực hiện các yêu cầu sau:

1- Xác định các khía cạnh, đối tượng cho từng câu bằng cách xác định các từ khóa liên quan về chủ đề sách (nội dung, chất lượng, đóng gói, giao hàng, dịch vụ, giá cả), đưa ra tối đa 3 khía cạnh

2- Đưa ra một trong ba sắc thái: positive, negative, neutral cho từng khía cạnh đó.

- 3- Tổng hợp kết quả bằng trình bày ra một file csv gồm các cột: STT, Decisions
- 4- Cột "Decisions" bao gồm khía cạnh và sắc thái với cú pháp: "{"khía cạnh":  
""sắc thái""}"

+ **Kết quả:**

```
STT,Decisions
0, "{"chất lượng": "positive", "nội dung": "negative", "đóng gói": "positive"}"
1, "{"nội dung": "positive"}"
2, "{"nội dung": "positive", "chất lượng": "positive"}"
3, "{"nội dung": "neutral"}"
4, "{"giao hàng": "positive", "đóng gói": "positive"}"
5, "{"chất lượng": "positive", "giao hàng": "positive", "đóng gói": "positive"}"
6, "{"chất lượng": "positive", "giao hàng": "negative"}"
7, "{"giao hàng": "negative", "dịch vụ": "negative"}"
8, "{"giao hàng": "positive", "đóng gói": "positive", "chất lượng": "positive"}"
9, "{"giao hàng": "positive", "đóng gói": "positive"}"
10, "{"nội dung": "negative"}"
11, "{"nội dung": "positive", "chất lượng": "positive"}"
12, "{"giao hàng": "neutral", "đóng gói": "negative", "nội dung": "neutral"}"
13, "{"giao hàng": "negative", "đóng gói": "negative"}"
14, "{"giao hàng": "neutral", "đóng gói": "negative", "nội dung": "positive"}"
```

## Chương IV: Phương pháp giải quyết bài toán

### 1. Động lực lựa chọn phương pháp giải quyết

#### - *Crawl dữ liệu từ web:*

+ Hiện nay trên internet thì kho dữ liệu văn bản, web là nguồn tài nguyên dồi dào quý giá. Việc khai thác và sử dụng chúng một cách hiệu quả sẽ mang lại lợi ích to lớn cho doanh nghiệp.

+ Nhiều thư viện mã nguồn mở của các ngôn ngữ lập trình hỗ trợ cho việc crawl web như: BeautifulSoup, Selenium, Scrapy,...

- *Đánh nhãn dữ liệu:* Dựa trên phần đánh nhãn dữ liệu sử dụng ChatGPT.

#### - *Tiền xử lý dữ liệu:*

+ Dữ liệu bình luận của người dùng thường không quan tâm đến chữ hoa, chữ thường  $\Rightarrow$  Chuyển hết sang chữ thường để dễ dàng xử lý.

+ Loại bỏ những ký tự kéo dài (thường là những từ thể hiện rõ cảm xúc).

**Ví dụ:** Sách hay quáaaaaaaa  $\rightarrow$  Sách hay quá.

+ Chuẩn hóa một số thuật ngữ khó hiểu, các từ viết tắt, ...

+ Loại bỏ các ký tự đặc biệt (dấu câu, các ký tự nhiễu, ...)

+ Sửa lỗi chính tả.

+ Sử dụng các công cụ tách từ Tiếng Việt (Pyvi, Underthesea, ...).

- *Biến đổi dữ liệu:* Trước khi đưa vào các mô hình huấn luyện, ta cần phải vector hóa từ dữ liệu dạng text sang dữ liệu số (các vector) bằng nhiều cách khác nhau: (Bag-of-Words, TF-IDF, Word2Vec, ...).

- **Xây dựng mô hình:**

- + Hiện nay có nhiều cách tiếp cận để giải quyết bài toán phân loại (Classification). Mỗi phương pháp có ưu, nhược điểm riêng phụ thuộc vào dữ liệu và bài toán là phân loại 2 lớp hay đa lớp,...
- + Với phương pháp học máy truyền thống, có thể sử dụng các mô hình phân lớp như SVM, Naive Bayes,...
- + Với cách tiếp cận học sâu, ta cần bộ dữ liệu lớn, có thể sử dụng (RNN, LSTM, BERT)
- So sánh kết quả đạt được giữa các mô hình và đưa ra đánh giá, tinh chỉnh tham số mô hình cho phù hợp nhất.

## 2. Cơ sở lý thuyết áp dụng cho bài toán

### 2.1. TF-IDF (Term Frequency - Inverse Document Frequency)

Trong khai thác dữ liệu văn bản, phương pháp thống kê TF-IDF là phương pháp phổ biến nhất để xác định mức độ quan trọng của từ trong một đoạn văn bản. TF-IDF chuyển đổi dữ liệu dạng văn bản (text) thành dạng không gian vector. Nó bao gồm hai thành phần chính là TF và IDF.

**TF (Term Frequency)** là số lần xuất hiện của một từ trong văn bản. TF được tính bởi công thức:

$$tf(t, d) = \frac{f(t, d)}{T}, \text{ với:}$$

- +  $f(t, d)$ : số lần xuất hiện từ  $t$  trong văn bản  $d$ .
- +  $T$ : Tổng số từ trong văn bản đó.
- +  $tf(t, d) \in [0, 1]$ .

**IDF (Inverse Document Frequency)**: tính toán độ quan trọng của một từ. Khi tính toán TF, mỗi từ đều quan trọng như nhau, nhưng có một số từ trong tiếng Anh như "is", "of", "that", ... xuất hiện khá nhiều nhưng lại rất ít quan trọng. Vì vậy, ta cần một phương thức



bù trừ những từ xuất hiện nhiều lần và tăng độ quan trọng của những từ ít xuất hiện những có ý nghĩa đặc biệt cho một số đoạn văn bản hơn bằng cách tính IDF:

$$idf(t) = \log \frac{|D|}{|\{d \in D: t \in d\}|} \text{ với:}$$

+  $|D|$  : Tổng số văn bản trong tập D

+  $|\{d \in D: t \in d\}|$ : Số văn bản chứa từ t.

Tính IDF để giảm giá trị của những từ phổ biến. Mỗi từ chỉ có 1 giá trị IDF duy nhất trong tập văn bản.

**TF-IDF:**  $tfidf(t, d, D) = tf(t, d) \times idf(t, D)$

Những từ có giá trị TF-IDF cao là những từ xuất hiện nhiều trong văn bản và xuất hiện ít đối với các văn bản khác => Lọc ra những từ phổ biến, giữ lại những từ có giá trị cao.

- Ví dụ code Python về phương pháp Tf-idf:

```
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
documents = ["This is the first document.",
             "This document is the second document.",
             "And this is the third one.",
             "Is this the first document?"]
vectorizer = TfidfVectorizer()
tfidf_matrix = vectorizer.fit_transform(documents)
# In ra chỉ số của từ vựng
print("Vocabulary: ", vectorizer.vocabulary_)
print("TF-IDF Matrix: ")
print(tfidf_matrix.toarray())
```

- **Kết quả:**

```
Vocabulary: {'this': 8, 'is': 3, 'the': 6, 'first': 2, 'document': 1, 'second': 5, 'and': 0,
             'third': 7, 'one': 4}
```

TF-IDF Matrix:

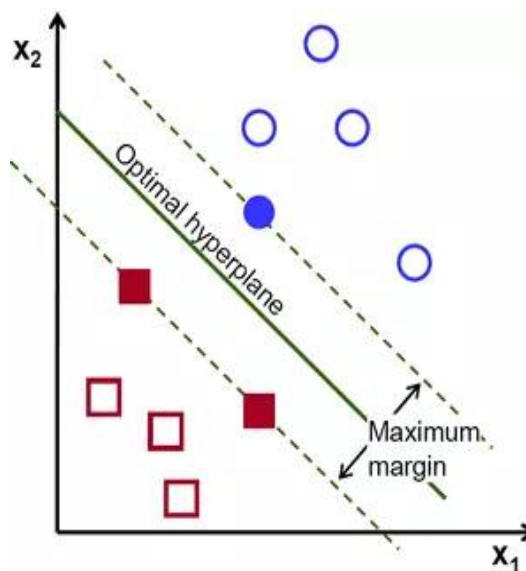
```
[[0 0.47 0.58 0.38 0 0 0.38 0 0.38]
 [0 0.69 0 0.28 0 0.54 0.28 0 0.28]
 [0.51 0 0 0.27 0.51 0 0.27 0.51 0.27]
 [0 0.47 0.58 0.38 0 0 0.38 0 0.38]]
```

## 2.2. SVM (Support Vector Machine)

SVM là một thuật toán học máy có giám sát được sử dụng trong bài toán phân loại 2 lớp hoặc nhiều lớp.

Ý tưởng của SVM là tìm một siêu phẳng (hyperlane) trong một không gian để phân chia các điểm dữ liệu thành các lớp khác nhau.

Margin là khoảng cách giữa siêu phẳng đến 2 điểm dữ liệu gần nhất tương ứng với các lớp. SVM luôn cố gắng cực đại hóa margin này, từ đó thu được một siêu phẳng tạo khoảng cách đều so với 2 nhãn.



Hình 4.1. Minh họa phân lớp SVM

- Ví dụ code Python minh họa thuật toán SVM:

```

from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score

iris = datasets.load_iris()
X_train,X_test,y_train,y_test = train_test_split(iris.data, iris.target, test_size=0.3,
random_state=0)

svm = SVC(kernel='linear', C=1, random_state=0)
svm.fit(X_train, y_train)
y_pred = svm.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

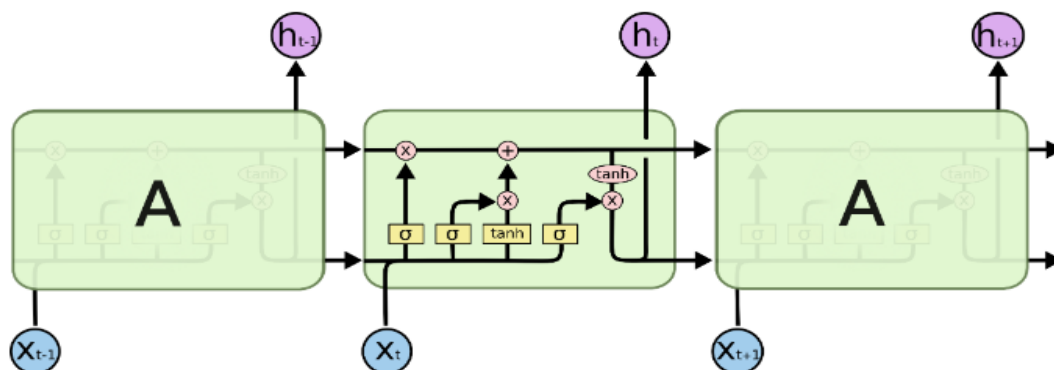
```

- **Kết quả:**

Accuracy: 0.977

## 2.3. LSTM (Long Short Term Memory)

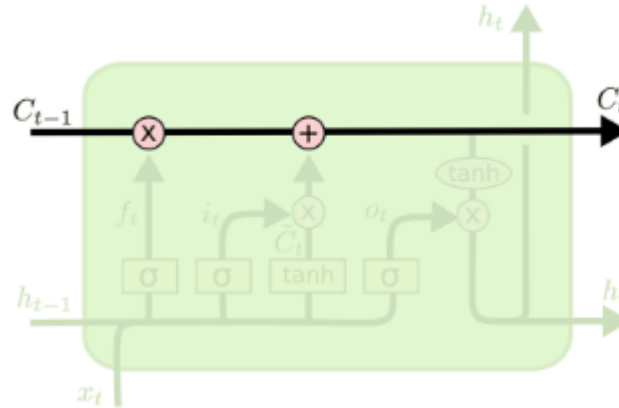
LSTM là mô hình cải tiến so với RNN. Khác với RNN chỉ học được những từ ở gần thì LSTM có khả năng học được những từ ở xa. LSTM có kiến trúc dạng chuỗi, thay vì chỉ có một tầng mạng nơ-ron của RNN, LSTM có tới 4 tầng tương tác với nhau.



Hình 4.2. Kiến trúc mạng LSTM

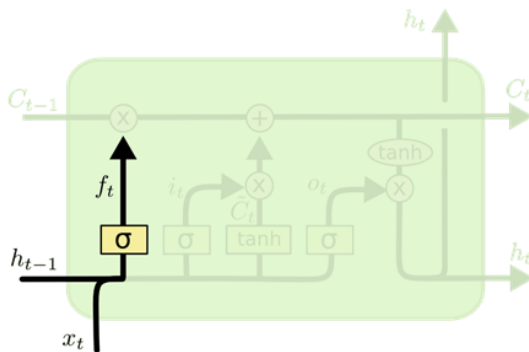
**\* Ý tưởng cốt lõi của LSTM:**

Chìa khóa của LSTM là trạng thái tế bào (cell state), nó chạy xuyên suốt tất cả các nút mạng và chỉ tương tác tuyến tính một chút. Thông tin được truyền đi một cách dễ dàng và không bị mất mát.



Hình 4.3. Cell state trong LSTM

Bước đầu tiên của LSTM là quyết định xem thông tin nào cần bỏ đi ở trạng thái tế bào. Quyết định này đưa ra bởi tầng sigmoid - tầng cổng quên (forget gate layer). Nó lấy đầu vào là  $h_{t-1}$  và  $x_t$  rồi trả về kết quả là một số trong khoảng  $[0,1]$  cho mỗi số trong trạng thái tế bào  $C_{t-1}$ . Đầu ra là 1 thể hiện rằng nó giữ toàn bộ thông tin lại, còn 0 chỉ rằng toàn bộ thông tin sẽ bị bỏ đi.

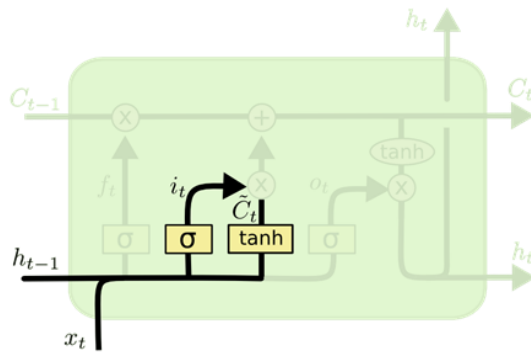


$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

Hình 4.4. Forget gate trong LSTM

Bước tiếp theo là lúc quyết định xem thông tin mới nào sẽ được lưu vào trạng thái tế bào. Công việc này bao gồm 2 phần. Đầu tiên là sử dụng một tầng Sigmoid được gọi là tầng cổng vào (input gate layer) để quyết định xem giá trị nào ta sẽ cập nhật.

Tiếp đó là một tầng tanh tạo ra một vector giá trị mới  $C_t$  nhằm thêm vào cho trạng thái. Kết hợp 2 phần trên để tạo ra một cập nhật mới cho trạng thái.

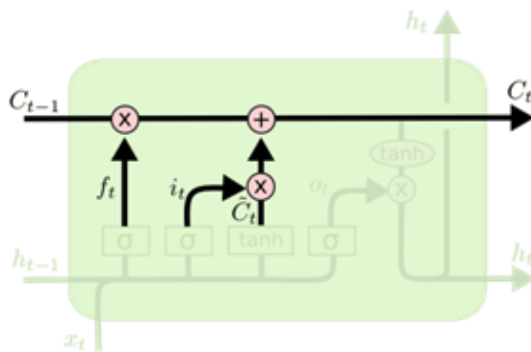


$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

Hình 4.5. Input gate trong LSTM

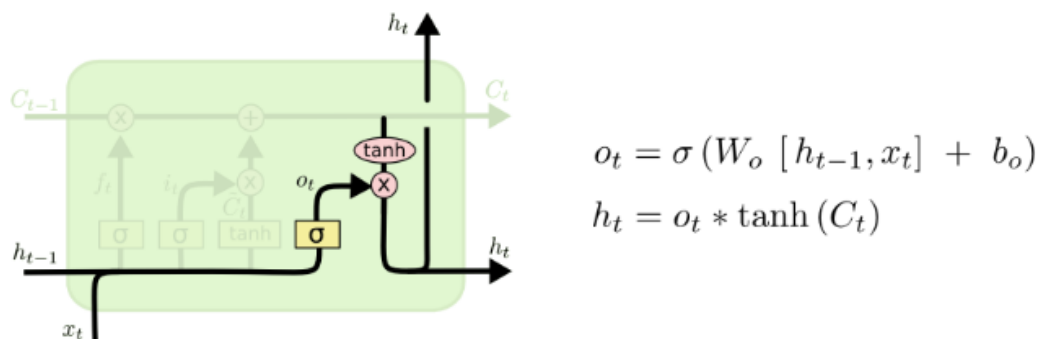
Tiếp theo ta cập nhật trạng thái tế bào cũ  $C_{t-1}$  thành trạng thái mới  $C_t$ . Nhân trạng thái cũ với  $f_t$  để bỏ đi những thông tin quyết định quên lúc trước, sau đó cộng thêm với  $i_t * \tilde{C}_t$ . Trạng thái mới này phụ thuộc vào việc quyết định cập nhật mỗi giá trị trạng thái.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Hình 4.6. Cập nhật trạng thái trong LSTM

Cuối cùng, cần quyết định xem mong muốn đầu ra là gì. Giá trị đầu ra sẽ dựa vào trạng thái tế bào, nhưng sẽ được tiếp tục sàng lọc. Đầu tiên, chạy một tầng sigmoid để quyết định phần nào của trạng thái tế bào ta muốn xuất ra. Sau đó, đưa nó trạng thái tế bào qua một hàm tanh để có giá trị nó về khoảng  $[-1, 1]$ , và nhân nó với đầu ra của cổng sigmoid để được giá trị đầu ra ta mong muốn.



Hình 4.7. Output gate trong LSTM

- Ví dụ code Python về LSTM

```
import numpy as np

from keras.datasets import imdb

from keras.models import Sequential

from keras.layers import Dense, LSTM, Embedding

from keras.utils import pad_sequence

max_features = 20000

maxlen = 80

batch_size = 32
```

```
(X_train, y_train), (X_test, y_test) = imdb.load_data(num_words=max_features)

X_train = sequence.pad_sequences(X_train, maxlen=maxlen)
X_test = sequence.pad_sequences(X_test, maxlen=maxlen)

model = Sequential()

model.add(Embedding(max_features, 128, input_length=maxlen))

model.add(LSTM(128, dropout=0.2, recurrent_dropout=0.2))

model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='adam',
metrics=['accuracy'])

model.fit(X_train, y_train, batch_size=batch_size, epochs=10,
validation_data=(X_test, y_test))

score, acc = model.evaluate(X_test, y_test, batch_size=batch_size)

print("Test score:", score)

print("Test accuracy:", acc)
```

**- Kết quả:**

```
Epoch 1/10
782/782 [=====] - 358s 447ms/step - loss:
0.4386 - accuracy: 0.7912 - val_loss: 0.3552 - val_accuracy: 0.8447

Epoch 2/10
782/782 [=====] - 327s 418ms/step - loss:
0.2537 - accuracy: 0.8982 - val_loss: 0.4840 - val_accuracy: 0.8208
```

Epoch 3/10

782/782 [=====] - 321s 410ms/step - loss:  
0.1628 - accuracy: 0.9398 - val\_loss: 0.4835 - val\_accuracy: 0.8253

Epoch 4/10

782/782 [=====] - 320s 409ms/step - loss:  
0.1077 - accuracy: 0.9598 - val\_loss: 0.5325 - val\_accuracy: 0.8158

Epoch 5/10

782/782 [=====] - 313s 400ms/step - loss:  
0.0713 - accuracy: 0.9745 - val\_loss: 0.7096 - val\_accuracy: 0.8076

Epoch 6/10

782/782 [=====] - 327s 419ms/step - loss:  
0.0459 - accuracy: 0.9852 - val\_loss: 0.7992 - val\_accuracy: 0.8143

Epoch 7/10

782/782 [=====] - 314s 401ms/step - loss:  
0.0449 - accuracy: 0.9852 - val\_loss: 0.8138 - val\_accuracy: 0.8156

Epoch 8/10

782/782 [=====] - 309s 395ms/step - loss:  
0.0338 - accuracy: 0.9888 - val\_loss: 0.9299 - val\_accuracy: 0.8058

Epoch 9/10

782/782 [=====] - 310s 396ms/step - loss:  
0.0259 - accuracy: 0.9924 - val\_loss: 0.9660 - val\_accuracy: 0.8179

Epoch 10/10



```
782/782 [=====] - 309s 395ms/step - loss:
0.0134 - accuracy: 0.9961 - val_loss: 0.9003 - val_accuracy: 0.8113

782/782 [=====] - 21s 27ms/step - loss:
0.9003 - accuracy: 0.8113

Test score: 0.9003129005432129

Test accuracy: 0.8113200068473816
```

## 2.4. PhoBERT

“Pretrained” PhoBERT models are the state of the art language models in Vietnamese (Pho, i.e “Phở”, is a popular food in VietNam)”. Đây là một pre-trained model được huấn luyện dành riêng cho tiếng Việt, được phát triển bởi cộng đồng nghiên cứu sinh VinAI.

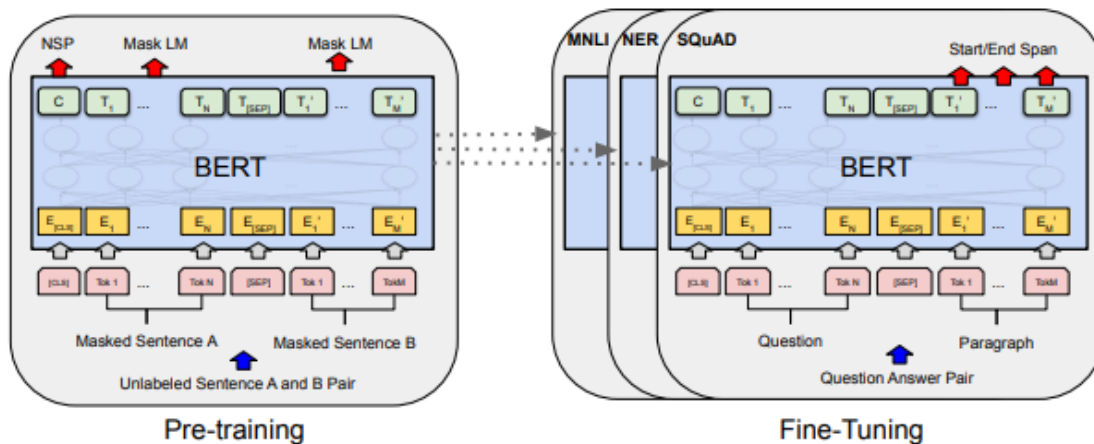
PhoBERT thừa hưởng mọi ưu điểm của BERT:

- Giữ được thông tin của ngữ cảnh trong quá trình huấn luyện và trích chọn đặc trưng nhận thông tin theo cả 2 chiều.
- Có thể huấn luyện với các token một cách độc lập giúp tăng tốc độ huấn luyện so với các mô hình khác.

Bên cạnh đó, PhoBERT còn khắc phục được các nhược điểm của BERT với Tiếng Việt:

- Trước PhoBERT, các mô hình BERT based không phân biệt được giữa tiếng và từ trong tiếng Việt.
- Tổng quan về BERT (Bidirectional Encoder Representation from Transformer) là mô hình biểu diễn từ hai chiều sử dụng kỹ thuật Transformer. BERT được thiết kế để pre-train word embedding, và đặc trưng của nó là có khả năng cân bằng bối cảnh cả hai chiều của câu. Cơ chế attention của Transformer cho phép mô hình học bối cảnh của từ dựa trên toàn bộ các từ trong câu, bao gồm cả bên trái và bên phải. Khác với các mô hình

embedding trước đó, BERT có thể được fine-tuning cho các tác vụ cụ thể bằng cách thêm một output layer vào kiến trúc của nó. Mặc dù thực tế là Transformer không hoàn toàn huấn luyện hai chiều, nhưng nó cho phép mô hình học được mối quan hệ giữa các từ trong câu đồng thời và không phụ thuộc vào chiều của câu.



Hình 4.8. Các tác vụ trong mô hình BERT

- Sử dụng model RoBERTa để thay thế cho mô hình BERT.

- **Kiến trúc của RoBERTa:**

+ RoBERTa được cải tiến hơn so với BERT bằng cách được huấn luyện trên mô hình lớn hơn, nhiều batch dữ liệu hơn, loại bỏ việc tìm câu kế tiếp (NSP) và thay đổi mặt nạ trong khi huấn luyện mô hình.

+ Kiến trúc của PhoBERT giống với RoBERTa bao gồm các lớp Encoder và Decoder của Transformer nằm chồng lên nhau.

- **Mã hóa BPE (Byte Pair Encoding)**

+ BPE là một phương pháp nén từ cơ bản, cho phép chỉ mục hóa toàn bộ các từ bao gồm cả những từ không có trong từ điển bằng cách mã hóa chúng thành chuỗi các từ phụ.

+ Subword được hiểu là từ được mã hóa lớn hơn kí tự và nhỏ hơn cấp độ từ. Ví dụ như từ lower có thể tách ra làm 2 thành phần từ phụ là low và er.

+ BPE đếm số lần xuất hiện của các từ phụ và tìm cách ghép chúng lại nếu chúng có tần suất xuất hiện cao nhất. Tiếp tục quá trình ghép từ phụ cho đến khi không còn subword nào có thể ghép được nữa, ta thu được một tập hợp subwords cho toàn bộ văn bản, trong đó mọi từ đều có thể được biểu diễn thông qua subwords này.

**- *Kiến trúc mô hình phân loại:***

+ Model BERT base sẽ tạo ra một kiến trúc gồm 12 sub-layers ở encoder, 12 heads trong multi-head attention trên mỗi sublayer. Output là tập hợp các vector self-attention bằng chiều dài của input. Mỗi vector có kích thước là 768.

+ Để điều chỉnh lại kiến trúc BERT cho tác vụ phân loại, ta cắt bỏ phần decoder của BERT, giữ lại phần encoder của transformer và sau đó trích xuất biểu diễn vector của token CLS đánh dấu vị trí đầu tiên. Để thực hiện phân loại, vector này sẽ được đưa vào mô hình thông qua việc thêm một lớp chiếu tuyến tính (fully connected layer) cuối cùng với kích thước bằng số lượng lớp cần phân loại.

## Chương V: Các bước giải quyết bài toán

### 1. Crawl dữ liệu

- Download chromedriver để kết nối và điều khiển trình duyệt Google Chrome từ một ứng dụng trình điều khiển hoặc một script tự động hóa (ở đây sử dụng selenium).
- Cài đặt driver sử dụng thư viện selenium trong python:

```
from selenium import webdriver

from selenium.webdriver.chrome.service import Service

chrome_driver_path='B:\\Q.MinhHust\\Project\\TikiSentimentAnalysis\\chromedriver.exe'

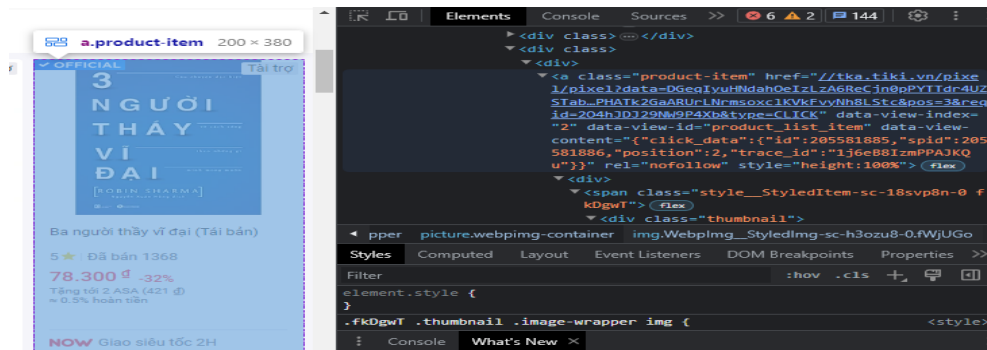
chrome_service = Service(chrome_driver_path, options = chrome_options)

driver = webdriver.Chrome(service=chrome_service)
```

- Để crawl dữ liệu từ web, bóc từng thuộc tính (các thẻ HTML) để đọc dữ liệu, dựa vào tên thẻ và các class, id để chỉ định dữ liệu crawl.
- URL để crawl:

```
https://tiki.vn/nha-sach-
tiki/c8322?src=c.8322.hamburger_menu_fly_out_banner&page=
```

- Lấy link của từng item

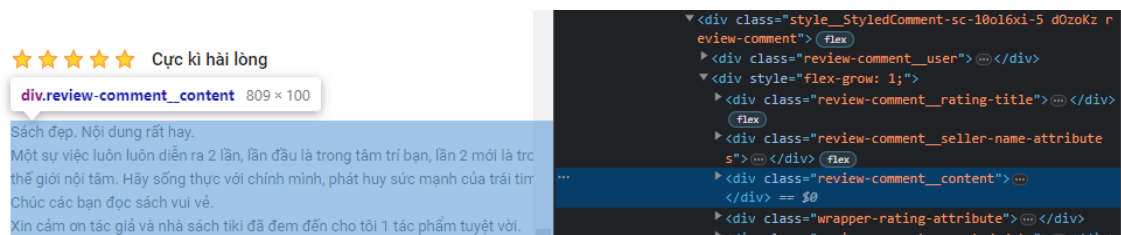


Hình 5.1. Crawl link

- + Bóc tách thuộc tính href trong thẻ có class = “product-item” để lấy link của item.
- + Để lấy trong nhiều page thì dùng vòng for để chạy các giá trị của page từ 1 đến 50 (vì chỉ có tối đa 50 trang).

```
url="https://tiki.vn/nha-sach-
tiki/c8322?src=c.8322.hamburger_menu_fly_out_banner&page="
links = []
for i in range(1, 50):
    driver.get(url + str(i))
    driver.execute_script("window.scrollTo(0,
    document.body.scrollHeight)")
    time.sleep(10)
    elem = driver.find_elements(By.XPATH, "//a[@class='product-item']")
    lt = []
    lt = [el for el in elem]
    for l in lt:
        links.append(l.get_attribute('href'))
```

- Lấy comment của từng product: Sau khi đã lấy được toàn bộ link của các item, ta truy cập lần lượt từng link một để lấy bình luận của người dùng.



Hình 5.2. Crawl comment

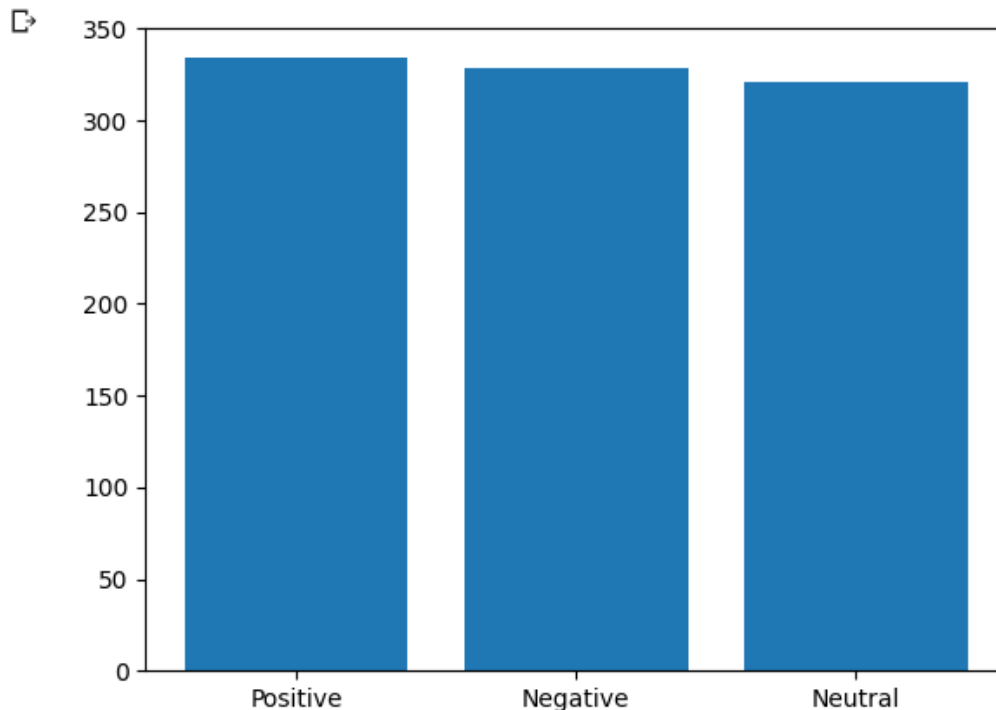
```
for link in len(links):
    driver.get(link)
    driver.execute_script("window.scrollTo(0,document.body.scrollHeight)")
    time.sleep(10)
elements=driver.find_elements(By.XPATH,"//*[@class='review-
comment__content']")
comments = [el.text for el in elements]
```

## 2. Tổng quan về bộ dữ liệu

### 2.1. Dữ liệu cho bài toán phân tích sắc thái cấp độ câu

- Sau khi crawl dữ liệu về xong, trích ra khoảng 984 câu bình luận phù hợp nhất để tạo bộ dữ liệu.
- Dữ liệu được gán cho 3 loại nhãn: các câu bình luận tích cực được đánh là 1, tiêu cực là -1 và trung lập là 0.
- Tổng quan về bộ dữ liệu sau khi được ChatGPT đánh nhãn:

```
data_normalized['sentiment'].value_counts()
-1    334
0     329
1     321
Name: sentiment, dtype: int64
```



Hình 5.3. Tổng quan bộ dữ liệu

Ta có thể thấy dữ liệu thu thập được rất cân bằng.

## 2.2. Dữ liệu cho bài toán phân tích sắc thái theo khía cạnh

- Sau khi crawl dữ liệu về xong, trích ra khoảng 984 câu bình luận phù hợp nhất để tạo bộ dữ liệu.

- Mỗi câu bình luận chỉ bao gồm 1-3 cặp (khía cạnh, sắc thái).

+ 6 khía cạnh xét tới là: nội dung, chất lượng, giao hàng, đóng gói, giá cả, dịch vụ.

+ 3 sắc thái được nhắc tới là tích cực, tiêu cực và trung lập.

⇒ Có tổng cộng 18 cặp (khía cạnh, sắc thái) có thể tạo ra trên toàn bộ tập dữ liệu.

- Tổng quan về số lượng cặp (khía cạnh, sắc thái).

Cặp khía cạnh, sắc thái	Label	Số lượng
(nội dung,positive)	1	279
(giao hàng,positive)	10	233
(chất lượng,positive)	4	179
(chất lượng,negative)	5	162
(đóng gói,negative)	8	136
(đóng gói,positive)	7	121
(nội dung,neutral)	3	86
(chất lượng,neutral)	6	73
(giao hàng,negative)	11	61
(nội dung,negative)	2	50
(dịch vụ,negative)	14	38
(giá cả,positive)	16	36
(dịch vụ,positive)	13	24
(giao hàng,neutral)	12	22
(đóng gói,neutral)	9	15
(giá cả,negative)	17	9
(dịch vụ,neutral)	15	9
(giá cả,neutral)	18	8

Bảng 2: Dữ liệu đánh nhãn theo khía cạnh

### 3. Tiền xử lý dữ liệu

- Chuyển hết các câu bình luận sang chữ thường:

```
mess_unic = convert_unicode(mess).lower()
```

- Loại bỏ các ký tự đặc biệt, chữ số.



```
mess_rmspectoken=re.findall(r'(?i)\b[a-  
zâãäåæçèéêëìíîïðñòóôõöøùúûüýÿđ0-  
9]+\b', mess_unic)
```

- Loại bỏ các kí tự kéo dài.

```
mess_norm = re.sub(r'([A-Z])\1+', lambda m: m.group(1).lower(), mess_norm,
flags = re.IGNORECASE)
```

- Chuẩn hóa thuật ngữ.

' k ': u' không ', ' bookmark ': u' dấu trang '

- Sử dụng thư viện Pyvi để thực hiện tách từ trong Tiếng Việt.

```
for i in range(len(data_normalized)):
    data_normalized['content'][i] =
    ViTokenizer.tokenize(data_normalized['content'][i])
```

**\* Ví dụ:** Câu ban đầu chưa tiền xử lý dữ liệu:

*‘Hình thức đẹp, còn nguyên màng co. Nội dung thì cũng là văn phong của NNA trước giờ chứ k mới mẻ đâu. Là mình thì thấy k hay như truyện Quán gò đi lên hay Nữ sinh. Mình sưu tầm và đọc tất cả sách NNA nên đáng mua lắm. Đặc biệt có vài câu thơ khá hay. Bookmark đẹp’*

⇒ Câu sau khi đã được tiền xử lý dữ liệu:

*‘hình\_thức đẹp còn nguyên màng co nội\_dung thì cũng là văn\_phong của nguyên nhật  
ánh trước giờ chứ không mới\_mẻ đâu là mình thì thấy không hay như truyện quán gò đi  
lên hay nữ\_sinh mình sưu\_tâm và đọc tất\_cả sách nguyên nhật ánh nên đánh mua lắm  
đặc\_biệt có vài câu thơ khá hay dấu trang đẹp’*

## 4. Xây dựng mô hình

### 4.1. SVM

#### - *Biến đổi dữ liệu:*

+ Sử dụng CountVectorizer để biến đổi string đầu vào thành các vector. Mỗi string sẽ chuyển thành một vector có độ dài d (số từ xuất hiện ít nhất 1 lần trong văn bản), giá trị của thành phần thứ i trong vector đó là số lần xuất hiện từ đó trong string.

+ Sử dụng TfidfTransformer để biến đổi ma trận vector vừa được vector hóa thành một ma trận đặc trưng Tf-idf.

```
steps.append(('CountVectorizer',CountVectorizer(ngram_range=(1,5),max_df=0.5, min_df=5)))
steps.append(('tfidf',TfidfTransformer(use_idf=False,sublinear_tf=True,norm='l2',smooth_idf=True)))
```

- Sử dụng GridSearchCV để lựa chọn các tham số phù hợp trong mô hình SVM.

```
grid.best_params_
⇒ Kết quả: {'C': 1, 'kernel': 'linear'}
```

- Lựa chọn tham số C = 1, kernel = linear.

```
steps.append(('classifier', SVC(C = 1, kernel = 'linear')))
```

### 4.2. LSTM

#### - *Biến đổi dữ liệu:*

+ Tìm độ dài từ lớn nhất trong 1 câu bình luận:

```
seq_len = [len(i.split()) for i in data_normalized.content]
print(max(seq_len))
⇒ Kết quả: 501
```

+ Tìm số từ vựng có trong từ điển:

```
word = set()
for text in data_normalized.content:
    for i in text.split(" "):
        word.add(i)
print(len(word))
⇒ Kết quả: 4675
```

+ Lựa chọn các tham số:

vocab\_size = 4500 (số từ vựng có trong từ điển).

max\_comment\_length = 500 (số từ vựng nhiều nhất có thể trong 1 câu).

embedding\_dims = 50 (số chiều của mỗi từ vựng trong không gian biểu diễn từ).

+ Chuyển văn bản đầu vào sang vector:

- Xác định số từ vựng tối đa cần biểu diễn.

```
tokenizer = Tokenizer(num_words=vocab_size, split = ' ')
```

- Xây dựng từ điển, mỗi từ được biểu diễn bởi một số nguyên duy nhất.

```
tokenizer.fit_on_texts(X)
```

- Mã hóa các từ thành các chuỗi số nguyên.

```
X = tokenizer.texts_to_sequences(X)
```

- Đưa các chuỗi số nguyên về cùng một độ dài để chuẩn bị đưa vào mô hình.

```
X = pad_sequences(X, maxlen=max_comment_length)
```

- **Xây dựng mô hình:**

+ Khởi tạo mô hình:

```
model = Sequential()
```

+ Thêm lớp Embedding để biểu diễn từ trong không gian số chiều thấp.

```
model.add(Embedding(vocab_size, embedding_dims, input_length=X.shape[1]))
```

+ Thêm lớp SpatialDropout1D để giảm overfitting.

```
model.add(SpatialDropout1D(0.2))
```

+ Thêm lớp LSTM với 100 đơn vị và các lớp dropout kèm theo với các lớp hiện tại và trước đó.

```
model.add(LSTM(100, dropout = 0.2, recurrent_dropout = 0.2))
```

+ Cuối cùng là lớp Dense với 3 lớp đầu ra để phân thành 3 lớp.

```
model.add(Dense(3, activation = 'softmax'))
```

+ Mô hình xây dựng được tổng hợp như sau:

Layer	Output shape	Param
embedding	(None, 500, 50)	225000
spatial_dropout1d	(None, 500, 50)	0
lstm	(None, 100)	60400
dense	(None, 3)	303
Tổng số params: 285703		

*Bảng 3: Thông số mô hình LSTM*

+ Cấu hình huấn luyện mô hình:

- Loss (hàm mất mát): 'categorical\_crossentropy' để phân loại đa lớp.
- Tối ưu hóa Adam.
- Độ đo: Accuracy (độ chính xác).

```
model.compile(loss = 'categorical_crossentropy', optimizer = 'adam', metrics = ['accuracy'])
```

+ Huấn luyện mô hình:

- Đầu vào là tập train đã được chia theo tỉ lệ 80%.
- epochs = 10 (số lần lặp lại toàn bộ tập dữ liệu).
- batch\_size = 32 (số lượng mẫu dữ liệu sử dụng trong mỗi lần lặp).
- validation\_data = 0.2 (trích ra 20% tập train để đánh giá hiệu suất mô hình).

```
model.fit(X_train, y_train, epochs = 10, batch_size = 32, validation_split = 0.2)
```

### 4.3. PhoBERT

- Download pretrained model PhoBERT:

```
!wget https://public.vinai.io/PhoBERT_base_fairseq.tar.gz  
!tar -xzf PhoBERT_base_fairseq.tar.gz
```

- Sau khi giải nén xong sẽ có 3 file:

+ bpe.codes: Là BPE token mà mô hình đã áp dụng để mã hóa văn bản sang index.

+ dict.txt: Bộ từ điển subword được sử dụng trong quá trình huấn luyện.

+ model.pt: Mô hình trên PyTorch được lưu trữ.

- Phân chia tập train (huấn luyện) và tập valid (kiểm thử):

```
content_train,content_val,label_train,label_val=train_test_split(data_normalized['content'], data_normalized['sentiment'], test_size = 0.2)
```

- Khởi tạo bộ mã hóa BPE:

```
parser = argparse.ArgumentParser()
parser.add_argument('--bpe-codes',
                    default="PhoBERT_base_transformers/bpe.codes",
                    required=False, type=str, help='path to fastBPE BPE')
args, unknown = parser.parse_known_args()
bpe = fastBPE(args)
```

- Khởi tạo từ điển:

```
vocab = Dictionary()
vocab.add_from_file("PhoBERT_base_transformers/dict.txt")
```

- Mã hóa BPE các tập train, valid với max\_len = 200 đưa các câu về cùng độ dài:

```
train_ids = []
for sent in content_train:
    subwords = '<s>' + bpe.encode(sent) + '</s>'
    encoded_sent=vocab.encode_line(subwords,append_eos=True,add_if_not_exist=False).long().tolist()
    train_ids.append(encoded_sent)
train_ids = pad_sequences(train_ids, maxlen=MAX_LEN, dtype="long", value=0,
                          truncating="post", padding="post")

val_ids = []
for sent in content_val:
    subwords = '<s>' + bpe.encode(sent) + '</s>'
    encoded_sent=vocab.encode_line(subwords,append_eos=True,
    add_if_not_exist=False).long().tolist()
    val_ids.append(encoded_sent)
```

```
val_ids=pad_sequences(val_ids,maxlen=MAX_LEN,dtype="long",value=0,truncating="post", padding="post")
```

- Tạo mask cho các câu:

```
train_masks = []
for sent in train_ids:
    mask = [int(token_id > 0) for token_id in sent]
    train_masks.append(mask)
val_masks = []
for sent in val_ids:
    mask = [int(token_id > 0) for token_id in sent]
    val_masks.append(mask)
```

- Chuyển hết các tập train, valid, test sang tensor để chuẩn bị đưa vào Pytorch:

```
train_inputs = torch.tensor(train_ids)
val_inputs = torch.tensor(val_ids)
train_labels = torch.tensor(label_train.values)
val_labels = torch.tensor(label_val.values)
train_masks = torch.tensor(train_masks)
val_masks = torch.tensor(val_masks)
```

- Khởi tạo DataLoader:

```
train_data = TensorDataset(train_inputs, train_masks, train_labels)
train_sampler = SequentialSampler(train_data)
train_dataloader=DataLoader(train_data,sampler=train_sampler, batch_size=8)
val_data = TensorDataset(val_inputs, val_masks, val_labels)
val_sampler = SequentialSampler(val_data)
val_dataloader = DataLoader(val_data, sampler=val_sampler, batch_size=8)
```

- Load model RobertaForSequenceClassification để gọi pretrain model PhoBERT:

```
config=RobertaConfig.from_pretrained("PhoBERT_base_transformers/config.json",
from_tf=False, num_labels = 3, output_hidden_states=False)
BERT_SA=RobertaForSequenceClassification.from_pretrained("PhoBERT_base_tr
ansformers/model.bin", config=config)
```

- Huấn luyện mô hình:

+ epochs = 10.

+ Thiết bị sử dụng để huấn luyện là cuda.

+ Tối ưu hóa AdamW.

```
optimizer_grouped_parameters = [{'params': [p for n, p in param_optimizer if
not any(nd in n for nd in no_decay)], 'weight_decay': 0.01 },
{'params': [p for n, p in param_optimizer if any(nd in n for nd in no_decay)],
'weight_decay': 0.0}]
optimizer = AdamW(optimizer_grouped_parameters, lr=1e-5,
correct_bias=False)
```

+ Huấn luyện trên từng epoch, mỗi epoch lấy batch\_size = 8:

● Đưa dữ liệu đầu vào (input\_ids, input\_mask, labels) vào device:

```
b_input_ids = batch[0].to(device)
b_input_mask = batch[1].to(device)
b_labels = batch[2].to(device)
```

● Xóa bộ nhớ đệm của mô hình:

```
BERT_SA.zero_grad()
```

● Chạy mô hình với đầu vào là input\_ids, input\_mask và labels, lấy ra giá trị loss:



```
outputs = BERT_SA(b_input_ids,
                  token_type_ids=None,
                  attention_mask=b_input_mask,
                  labels=b_labels)
loss = outputs[0]
total_loss += loss.item()
```

- Tính toán độ chính xác và F1-score của các dự đoán được tạo ra bởi mô hình:

```
logits = outputs[1].detach().cpu().numpy()
label_ids = b_labels.to('cpu').numpy()
tmp_train_accuracy, tmp_train_f1 = flat_accuracy(logits, label_ids)
train_accuracy += tmp_train_accuracy
train_f1 += tmp_train_f1
```

- Cập nhật giá trị của các tham số trong mô hình:

```
loss.backward()
torch.nn.utils.clip_grad_norm_(BERT_SA.parameters(), 1.0)
optimizer.step()
```

- Đưa ra accuracy, f1 score và hàm mất mát trung bình trên từng epoch:

```
print(" Accuracy: {0:.4f}".format(train_accuracy/nb_train_steps))
print(" F1 score: {0:.4f}".format(train_f1/nb_train_steps))
print(" Average training loss: {0:.4f}".format(avg_train_loss))
```

+ Đánh giá mô hình huấn luyện trên tập valid:

```
with torch.no_grad():
    outputs = BERT_SA(b_input_ids,
                    token_type_ids=None,
                    attention_mask=b_input_mask)
```

```
logits = outputs[0]
logits = logits.detach().cpu().numpy()
label_ids = b_labels.to('cpu').numpy()
tmp_eval_accuracy, tmp_eval_f1 = flat_accuracy(logits, label_ids)
eval_accuracy += tmp_eval_accuracy
eval_f1 += tmp_eval_f1
nb_eval_steps += 1
```

## 5. Đánh giá kết quả

Mô hình	Accuracy	F1 score	Đánh giá
SVM	0.69	0.68	Tốt
LSTM	0.93	0.59	Chưa tốt
PhoBERT	0.97	0.96	Rất tốt

Bảng 4: So sánh các mô hình

## KẾT LUẬN

Đồ án đã hoàn thành mục tiêu đề ra: “Ứng dụng ChatGPT trong bài toán phân tích sắc thái bình luận sách”

Trong đồ án này, em đã tìm hiểu tổng quan về ChatGPT và cách ứng dụng nó trong việc đánh nhãn dữ liệu đối với bài toán phân tích sắc thái. Bên cạnh em cũng thực hiện crawl dữ liệu từ web và xây dựng được 3 mô hình SVM, LSTM và PhoBERT với độ chính xác khá tốt.

Tuy nhiên mô hình LSTM của em vẫn còn chưa tốt ở metric f1-score (0.6), trong tương lai em sẽ thực hiện tiền xử lý văn bản tốt hơn và dùng các phương pháp tinh chỉnh siêu tham số để cải thiện mô hình này.

## TÀI LIỆU THAM KHẢO

- [1] Hồ Ngọc Hải, “ChatGPT: The Evolution of Natural Language Processing”.
- [2] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever – “Improving Language Understanding by Generative Pre-Training”.
- [3] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever – “Language Models are Unsupervised Multitask Learners”.
- [4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, Dario Amodei – “Language Models are Few-Shot Learners”.
- [5] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, Ryan Lowe – “Training language models to follow instructions with human feedback”.
- [6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin – “Attention Is All You Need”.
- [7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova - BERT: “Pre-training of Deep Bidirectional Transformers for Language Understanding”.

- [8] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, Veselin Stoyanov – “RoBERTa: A Robustly Optimized BERT Pretraining Approach”
- [9] Dat Quoc Nguyen, Anh Tuan Nguyen - PhoBERT: “Pre-trained language models for Vietnamese”.
- [10] Mickel Hoang, Oskar Alija Bihorac Jacobo Rouces – “Aspect-Based Sentiment Analysis Using BERT”.