**BÁO CÁO CUỐI KỲ**

**Môn: Khai thác dữ liệu truyền thông xã hội**

**Đề tài: Bài toán phân lớp trong việc xác định hate speech của các bình luận trên mạng xã hội**

**Giảng viên hướng dẫn:** Ths. Nguyễn Văn Kiệt

**Lớp:** IE403.M21

**Nhóm sinh viên thực hiện:**

**Danh sách thành viên:**

|  |  |
| --- | --- |
| Mã số sinh viên | Họ và tên |
| 18521300 | Võ Thị Ngọc Ân |
| 17521319 | Huỳnh Bảo Minh |

*TP.HỒ CHÍ MINH , 9 tháng 5 năm 2022*

**Nhận xét của giảng viên**

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

|  |  |
| --- | --- |
|  | Thành phố Hồ Chí Minh, ngày …… tháng …… năm 2022 |
|  | **Chữ ký giảng viên** |
|  |  |
|  |  |

# 

**Mục lục**

[3](#_heading=h.gjdgxs)

[Danh mục ảnh 6](#_heading=h.30j0zll)

[Chương 1: Tổng quan 8](#_heading=h.1fob9te)

[1.](#_heading=h.3znysh7) Giới thiệu 8

[Tình hình xã hội 8](#_heading=h.2et92p0)

[Phát biểu bài toán 8](#_heading=h.tyjcwt)

[2.](#_heading=h.3dy6vkm) Đối tượng khai thác – Mục tiêu – Dataset 8

[Đối tượng khai thác 8](#_heading=h.1t3h5sf)

[Mục tiêu 8](#_heading=h.4d34og8)

[Dataset 8](#_heading=h.2s8eyo1)

[3.](#_heading=h.17dp8vu) Khó khăn 9

[Chương 2: Các công trình nghiên cứu liên quan 10](#_heading=h.3rdcrjn)

[Chương 3: Mô hình giải bài toán 11](#_heading=h.26in1rg)

[1.](#_heading=h.lnxbz9) Tiền xử lý dữ liệu 11

[2.](#_heading=h.3j2qqm3) Mô tả đặc trưng 13

[3.](#_heading=h.1y810tw) Phương pháp đề xuất 13

[**1)**](#_heading=h.4i7ojhp) **BERT** 13

[2)](#_heading=h.2xcytpi) Các độ đo cần quan tâm 14

[Chương 4: Cài đặt thực nghiệm 18](#_heading=h.1pxezwc)

[1.](#_heading=h.49x2ik5) Dữ liệu thực nghiệm 18

[2.](#_heading=h.ihv636) Phương pháp thực nghiệm 18

[3.](#_heading=h.41mghml) Kết quả thực nghiệm 19

[Chương 5: Demo chương trình 21](#_heading=h.1v1yuxt)

[1.](#_heading=h.4f1mdlm) Tải các thư viện cần thiết 21

[2.](#_heading=h.28h4qwu) Tiền xử lý dữ liệu 21

[3.](#_heading=h.46r0co2) Áp dụng mô hình PhoBERT 23

[4.](#_heading=h.3cqmetx) Áp dụng mô hình đã được train cho một bộ dữ liệu chưa có label 28

[5.](#_heading=h.1664s55) Áp dụng mô hình Naïve-Bayes 30

[Chương 6: Kết luận 33](#_heading=h.1jlao46)

[Tài liệu tham khảo 34](#_heading=h.43ky6rz)

Danh mục ảnh

[Hình 1: Bộ dữ liệu tổng thể 10](#_heading=h.35nkun2)

[Hình 2: biểu đồ phần trăm nhãn các comment 10](#_heading=h.1ksv4uv)

[Hình 3: wordcloud của tag clean 11](#_heading=h.44sinio)

[Hình 4: wordcloud của tag offensive 11](#_heading=h.2jxsxqh)

[Hình 5: wordcloud của tag hate 12](#_heading=h.z337ya)

[Hình 6: sơ đồ matrix confusion 14](#_heading=h.1ci93xb)

[Hình 7: công thức tính accuracy 14](#_heading=h.3whwml4)

[Hình 8: công thức tính precision 15](#_heading=h.2bn6wsx)

[Hình 9: công thức tính recall 15](#_heading=h.qsh70q)

[Hình 10: công thức tính F1 score 16](#_heading=h.3as4poj)

[Hình 11: xử lý tiền dữ liệu 17](#_heading=h.2p2csry)

[Hình 12: token hóa dữ liệu 17](#_heading=h.147n2zr)

[Hình 13: định nghĩa dataset mà mô hình sẽ sử dụng 17](#_heading=h.3o7alnk)

[Hình 14: đưa dữ liệu vào dataset được định nghĩa 17](#_heading=h.23ckvvd)

[Hình 15: huấn luyện mô hình bằng train, đánh giá bằng dev 18](#_heading=h.32hioqz)

[Hình 16: dự đoán label cho test 18](#_heading=h.1hmsyys)

[Hình 17: confusion matrix và hệ số F1 macro, micro 18](#_heading=h.2grqrue)

[Hình 18: biểu đồ nhiệt confusion matrix 18](#_heading=h.vx1227)

[Hình 19: dataset test với cột dự đoán 19](#_heading=h.3fwokq0)

[Hình 20: cài đặt thư viện transformer 20](#_heading=h.2u6wntf)

[Hình 21: cài đặt thư viện vncorenlp 20](#_heading=h.19c6y18)

[Hình 22: tải wordsegmenter 20](#_heading=h.3tbugp1)

[Hình 23: đọc dataset 20](#_heading=h.nmf14n)

[Hình 24: viết hàm tiền xử lý dữ liệu 21](#_heading=h.37m2jsg)

[Hình 25: tách dataset thành các cột dữ liệu riêng biệt để chuẩn bị bước tiền xử lý 22](#_heading=h.1mrcu09)

[Hình 26: gọi thư viện máy học 22](#_heading=h.2lwamvv)

[Hình 27: tiền xử lý dữ liệu và đưa vào dataset mà máy sẽ sử dụng 22](#_heading=h.111kx3o)

[Hình 28: viết các parameter cần thiết và tiến hành train model, sau đó lưu model 23](#_heading=h.3l18frh)

[Hình 29: nơi chứa model đã được train 23](#_heading=h.206ipza)

[Hình 30: nhận về kết quả dự đoán trên test và hiện các kết quả thu được 24](#_heading=h.4k668n3)

[Hình 31: kết quả hiện thị quá trình train 24](#_heading=h.2zbgiuw)

[Hình 32: kết quả khi model chạy trên test và các thông số độ đo cần lưu ý 25](#_heading=h.1egqt2p)

[Hình 33: vector dự đoán 25](#_heading=h.3ygebqi)

[Hình 34: đưa cột dự đoán vào dataset ban đầu để so sánh 26](#_heading=h.2dlolyb)

[Hình 35: đối chiếu các kết quả bị dự đoán sai 27](#_heading=h.sqyw64)

[Hình 36: tiền xử lý dữ liệu 28](#_heading=h.1rvwp1q)

[Hình 37: chuyển dự đoán của máy thành vector 29](#_heading=h.4bvk7pj)

[Hình 38: đưa vào dataset ban đầu để xem kết quả dự đoán của mô hình 29](#_heading=h.2r0uhxc)

[Hình 39: Chuyển hóa dữ liệu về dạng matrix và kết quả vector của matrix 30](#_heading=h.3q5sasy)

[Hình 40: Đưa dữ liệu vào mô hình Naive Bayes 30](#_heading=h.25b2l0r)

[Hình 41: Chạy kết quả và tính các độ đo cần thiết 30](#_heading=h.kgcv8k)

[Hình 42: Kết quả các độ đo cần lưu ý 31](#_heading=h.34g0dwd)

# Chương 1: Tổng quan

## Giới thiệu

### Tình hình xã hội

Hiện nay, có khoảng 70 triệu người dùng Internet tại Việt Nam, và hầu hết họ đều quen thuộc với Facebook và Youtube. Trên mạng xã hội, ngôn từ thù địch (hate speech) đã trở thành một vấn đề nhức nhối đối với người dùng mạng xã hội. Trên mạng xã hội, sự thù ghét dễ dàng xuất hiện và dễ dàng bị lan truyền. Lòng thù ghét dẫn đến sự hủy diệt nhân loại, cô lập con người và làm suy nhược xã hội. Trong sự phát triển của các trang mạng xã hội, sự căm ghét xuất hiện trên mạng xã hội dưới dạng các bình luận có lời lẽ căm thù, bài đăng hoặc tin nhắn có lời lẽ căm thù và nó lan truyền quá nhanh và quá dễ dàng. Sự tồn tại của ngôn từ thù địch khiến không gian mạng xã hội trở nên độc hại, đe dọa người dùng mạng xã hội và gây hoang mang cho cộng đồng.

Đã có nhiều bài viết, thông điệp, tweet,… với nỗ lực giáo dục, tuyên truyền nhằm ngăn chặn sự thù địch lan rộng của các tổ chức, cá nhân. Mặc dù những thông điệp tích cực chứng minh được độ ảnh hưởng, nhưng chừng đó vẫn chưa đủ để xoá bỏ thái độ tiêu cực, hoặc ít nhất là giảm nó đến mức không đáng kể. Do đó rất cần những biện pháp khác để hỗ trợ công việc này.

### Phát biểu bài toán

Bài báo cáo này ứng dụng bài toán phân lớp trên dữ liệu comment từ các trang mạng xã hội. Bộ dataset được sử dụng chứa hơn 30000 comment, mỗi comment được gán cho 1 trong 3 label: CLEAN, OFFENSIVE, HATE

Input: bộ dữ liệu chứa nội dung comment và label được gán

Output: label dự đoán của comment

## Đối tượng khai thác – Mục tiêu – Dataset

### Đối tượng khai thác

Các comment của người dùng mạng xã hội về các lĩnh vực như giải trí, người nổi tiếng, các vấn đề xã hội và chính trị từ các trang Facebook và các video trên Youtube. Lý do chọn 2 platform này là vì chúng có độ tương tác cao và không bị hạn chế comment. Sau đó, xóa tên người dùng để bảo đảm sự ẩn danh

### Mục tiêu

Giải bài toán phân lớp, tìm ra comment thuộc lớp CLEAN, , OFFENSIVE hoặc HATE

### Dataset

Được lấy từ trang github của thầy Lưu Thanh Sơn

Link dataset: <https://github.com/sonlam1102/vihsd/tree/main/data>

Dữ liệu được chia làm 3 phần: file train.csv để huấn luyện (24048 dòng), file dev.csv để đánh giá (2672 dòng) và file test để kiểm tra (6680 dòng)

Các cột trong dataset:

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Mô tả |
| free\_text | string | Bình luận của người dùng |
| label\_id | integer | Bình luận được gán nhãn dưới dạng số:  0 tương đương với CLEAN  1 tương đương với OFFENSIVE  2 tương đương với HATE |

## Khó khăn

* Là bài toán phân lớp nên yêu cầu độ chính xác cao.
* Sử dụng tài khoản miễn phí nên bị giới hạn dung lượng trong quá trình xử lý dữ liệu và huấn luyện mô hình.
* Thực hiện trên môi trường Google Colab nên đôi khi quá trình xử lý dữ liệu bị gián đoạn do đường truyền Internet thiếu ổn định.

# Chương 2: Các công trình nghiên cứu liên quan

Trong tiếng Anh và nhiều ngôn ngữ khác, có rất nhiều bộ dữ liệu được xây dựng để phục vụ nghiên cứu về vấn đề phát hiện hate speech. Các bộ dữ liệu này được chia làm 2 loại: flat labels và hierarchical labels. Flat label là các label được đánh giá là độc lập với nhau về mặt ý nghĩa. Ngược lại, hierarchical label là các label có thể có những label thuộc lớp con của những label khác hoặc các label này là một tập hợp chứa các label khác.

Đối với dataset loại flat label, có dataset của Waesem và Hovy chứa 17000 tweet trên Twitter với 3 nhãn: phân biệt chủng tộc (racism), phân biệt giới tính (sexism), không có (none)

Đối với dataset loại hierarchical label, Zampieri và đồng nghiệp cung cấp một bộ dữ liệu nhiều label để dự đoán các bài viết mang tính công kích trong tiếng Anh. Dataset này thực hiện 2 nhiệm vụ: công kích nhóm và công kích cá nhân với mỗi nhiệm vụ có 2 lớp nhãn

Với tiếng Việt:

Bộ dữ liệu VLSP-HSD của Vũ Xuân Sơn và các đồng nghiệp ở hội thảo VLSP 2019 về vấn đề phát hiện hate speech bằng tiếng Việt. Nhưng các tác giả không cung cấp quá trình xây dựng bộ dữ liệu cũng như phương thức đánh giá chất lượng bộ dữ liệu.

Bộ dữ liệu ViHSD của Lưu Thanh Sơn và các đồng nghiệp gồm 33400 comment từ các nền tảng mạng xã hội như Facebook, YouTube, Instagram. Các dòng dữ liệu được annotate bằng tay gồm 3 nhãn độc lập: CLEAN, OFFENSIVE, HATE

# Chương 3: Mô hình giải bài toán

## Tiền xử lý dữ liệu

Bộ dữ liệu đầy đủ bao gồm 33400 dòng

Table

Description automatically generated

*Hình 1: Bộ dữ liệu tổng thể*

Trước khi đến với các bước xử lý dữ liệu tiếp theo, nhóm thực hiện một số thao tác trực quan hoá để có cái nhìn cụ thể hơn về dữ liệu trước khi huấn luyện. Đầu tiên là vẽ một biểu đồ tròn cho biết tỉ lệ phân bố của các nhãn trong toàn tập dữ liệu.

Chart, pie chart

Description automatically generated

*Hình 2: biểu đồ phần trăm nhãn các comment*

Tiếp theo, dùng thư viện hỗ trợ wordcloud để xem những từ thường xuyên xuất hiện trong các comment là gì. Từ xuất hiện càng nhiều thì cỡ chữ càng to và nằm ở trung tâm. Kết quả này đóng góp một phần khá quan trọng trong việc nhận xét kết quả huấn luyện mô hình.

A close-up of a dollar bill

Description automatically generated with medium confidence

*Hình 3: wordcloud của tag clean*

A picture containing text

Description automatically generated

*Hình 4: wordcloud của tag offensive*

A close-up of a dollar bill

Description automatically generated with medium confidence

*Hình 5: wordcloud của tag hate*

Các bước xử lý tiền dữ liệu bao gồm:

* Lọc stop word: loại bỏ các stop word để mô hình học máy có thể tập trung vào những từ quan trọng mang ý nghĩa quyết định tới việc gắn label
* Lọc emoji: các emoji hiện tại không thể được máy phân tích cho nên chúng bị loại bỏ
* Chuyển đổi thành chữ in thường các kí tự nội dung: chuyển đổi các từ và các ký tự về chữ in thường để máy không phải tiếp nhận thêm các từ về bản chất là cùng nghĩa
* Token hóa chuỗi text: Đưa 1 câu comment hoàn chỉnh thành 1 list các từ vựng với mỗi 1 thành phần của list là 1 từ
* Normalizing: chuẩn hoá và đặt trọng số cho các token quan trọng

## Mô tả đặc trưng

Là bài toán phân lớp kết hợp xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP), chúng ta cần quan tâm đặc trưng được trích xuất từ text. Sau khi trải qua các bước tiền xử lý, bảng dữ liệu đã phù hợp cho việc huấn luyện mô hình, gồm một cột đặc trưng (cột free\_text sau khi trải qua các bước xử lý và trích xuất đã trở thành cột chứa các vector đặc trưng) và một cột nhãn dữ liệu (label\_id).

## Phương pháp đề xuất

### **BERT**

BERT - Bidirectional Encoder Representation from Transformer – là một mô hình biểu diễn ngôn ngữ được Google giới thiệu vào năm 2018. Tại thời điểm công bố, BERT đã tạo ra sự rung động trong cộng đồng NLP bởi những cải tiến chưa từng có ở những mô hình trước đó. Kể từ ngày đó, BERT vẫn chưa từng nguội đi sức nóng của mình và thực tế BERT ngày càng nóng hơn bao giờ hết. Với các phiên bản cải tiến, biến thể như RoBERTa, ALBERT, DistilBERT,... BERT đã càn quét các tác vụ xử lý ngôn ngữ tự nhiên, trở lên áp đảo trong các nền tảng thi đấu như Kaggle, AIVIVN cũng như shared task của nhiều hội nghị.

Trong bài báo BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, các tác giả đã nêu ra những cải tiến của mô hình BERT:

* Tăng GLUE score (General Language Understanding Evaluation score), một chỉ số tổng quát đánh giá mức độ hiểu ngôn ngữ lên 80.5%.
* Tăng điểm số accuracy trên bộ dữ liệu MultiNLI đánh giá tác vụ quan hệ văn bản (Text Entailment) lên 86.7%.
* Tăng điểm số accuracy, F1-score trên bộ dữ liệu SQuAD v1.1 đánh giá tác vụ Question and Answering lên 93.2%

Và ở Việt Nam, Viện nghiên cứu trí tuệ nhân tạo VinAI Research (thuộc Tập đoàn Vingroup) đã tạo ra một mô hình đã được qua huấn luyện chuyên dùng cho Tiếng Việt (Pre-trained language models for Vietnamese): PhoBERT (pho trong Tiếng Việt là phở - một loại đồ ăn nổi tiếng của Việt Nam với cộng đồng quốc tế). Nhóm đề xuất sử dụng PhoBert là công cụ để thực hiện việc phân lớp vì độ hiệu quả lớn và chuyên dụng cho Tiếng Việt của PhoBERT

Cuối cùng, vì BERT là một mô hình pre-trained, nên nó có một đặc điểm là cho phép tinh chỉnh tham số (fine-tune) để có thể đạt được mô hình SOTA (state-of-the-art) cho nhiều nhiệm vụ NLP khác nhau.

### Các độ đo cần quan tâm

#### Confusion Matrix

Confusion Matrix (gọi là Ma trận nhầm lẫn hay Ma trận lỗi) là một bố cục bảng cụ thể cho phép hình dung hiệu suất của một thuật toán, thường là một thuật toán có giám sát (trong học không giám sát thì thường được gọi là Matching Matrix – Ma trận phù hợp). Ma trận nhầm lẫn là một trong những kỹ thuật đo lường hiệu suất phổ biến nhất và được sử dụng rộng rãi cho các mô hình phân lớp. Mỗi dòng của ma trận đại diện cho số lượng dữ liệu thật sự thuộc về một lớp (lớp thực tế), trong khi mỗi cột đại diện cho số lượng dữ liệu mà mô hình dự đoán thuộc về lớp nào (lớp dự đoán). Tên gọi này bắt nguồn từ việc nó giúp chúng ta dễ dàng xem hệ thống bị nhầm lẫn giữa hai (hay nhiều) lớp nào (khi máy gán nhãn sai cho dữ liệu).

Table

Description automatically generated

*Hình 6: sơ đồ matrix confusion*

Từ cấu trúc của Confusion Matrix trên, ta có một số khái niệm sau:

* Condition Positive (P): số lượng thực tế dữ liệu có nhãn Positive
* Condition Negative (N): số lượng thực tế dữ liệu có nhãn Negative
* True Positive (TP): số lượng dữ liệu mà mô hình dự đoán đúng nhãn Positive
* True Negative (TN): số lượng dữ liệu mà mô hình dự đoán đúng nhãn Negative
* False Positive (FP): số lượng dữ liệu mà mô hình dự đoán là nhãn Positive nhưng nhãn thực tế không phải Positive
* False Negative (FN): số lượng dữ liệu mà mô hình dự đoán là nhãn Negative nhưng nhãn thực tế không phải Negative

Những giá trị này liên quan mật thiết đến nhiều công thức của các độ đo khác dùng để đánh giá mô hình phân lớp. Các độ đo liên quan đến Confusion Matrix sẽ được trình bày ở những phần tiếp theo.

#### Accuracy Score

Khi xây dựng mô hình phân loại chúng ta sẽ muốn biết một cách khái quát tỷ lệ các trường hợp được dự đoán đúng trên tổng số các trường hợp là bao nhiêu. Tỷ lệ đó được gọi là độ chính xác, hay Accuracy. Accuracy giúp ta đánh giá hiệu quả dự báo của mô hình trên một bộ dữ liệu. Accuracy càng cao thì mô hình của chúng ta càng chuẩn xác. Accuracy được ưa chuộng vì có thể nói nó là công thức dễ hiểu nhất trong các công thức liên quan đến Confusion Matrix, và nó cũng thể hiện khá chính xác hiệu suất mô hình. Công thức của Accuracy như sau:

Text

Description automatically generated with medium confidence

*Hình 7: công thức tính accuracy*

Tuy nhiên hạn chế của nó là đo lường trên tất cả các nhãn mà không quan tâm từng nhãn. Ví dụ có 10,000 bệnh nhân. Nhiệm vụ là phát hiện ra được những bệnh nhân có nguy cơ ung thư để điều trị càng sớm càng tốt cho họ. Mô hình dự đoán được 9,900 bệnh nhân có bệnh không nguy hiểm, nhưng không thể dự đoán được ca có nguy cơ ung thư nào. Vì Accuracy chỉ quan tâm số lượng nhãn dự đoán đúng, trong trường hợp này lên đến 99%, nhưng rõ ràng mô hình vẫn sai sót trường hợp có nguy cơ ung thư thật, dẫn đến bệnh nhân có nguy cơ không được điều trị kịp thời. Nghĩa là với nhiệm vụ này, chúng ta sẽ quan tâm độ chính xác đo lường chỉ trên nhãn “có nguy cơ ung thư” hơn. Khi đó chúng ta sẽ cần đến những độ đo khác như Precision hay Recall, sẽ được trình bày tiếp theo.

#### Precision

Precision là độ đo trả lời cho câu hỏi trong các trường hợp được dự đoán là Positive thì có bao nhiêu trường hợp đúng. Công thức Precision như sau:

Text

Description automatically generated with low confidence

*Hình 8: công thức tính precision*

Giá trị của Precision nằm trong khoảng 0 < Precision <= 1. Precision càng cao nghĩa là mô hình dự đoán chính xác nhiều dữ liệu có nhãn Positive. Precision được quan tâm hơn khi lựa chọn mô hình với các bài toán cụ thể khi mà việc nhận nhầm False Positive mang lại hậu quả. Ví dụ với bài toán xác định và chặn email spam, việc nhận nhầm False Positive (nhầm email thường thành email spam) có thể gây ảnh hưởng lớn đến người dùng (nhầm email giao dịch, email hợp đồng,… thành email spam).

#### Recall

Recall là độ đo dùng để đo lường tỉ lệ dự đoán chính xác các trường hợp Positive trên toàn bộ các mẫu thuộc nhóm Positive. Công thức Recall như sau:

Text, letter

Description automatically generated

*Hình 9: công thức tính recall*

Giá trị của Recall cũng nằm trong khoảng 0 < Recall <= 1. Recall càng cao nghĩa là tỉ lệ bỏ sót các dữ liệu nhãn Positive càng thấp. Do đó, Recall được quan tâm hơn khi lựa chọn mô hình với các bài toán cụ thể khi mà việc dự đoán True Positive nhầm thành False Negative mang lại hậu quả. Ví dụ như bài toán dự đoán ung thư trên, việc nhận nhầm bệnh nhân có nguy cơ ung thư thành bệnh không nguy hiểm để lại hậu quả rất lớn.

#### F1 Score

Khi huấn luyện mô hình, chúng ta đều mong muốn Precision và Recall đều cao. Tuy nhiên trong thực tế, kết quả của mô hình thường là Precision cao thì Recall sẽ thấp hơn, và ngược lại. Lúc đó rất khó để lựa chọn đâu là một mô hình tốt vì ta sẽ phân vân không biết đánh giá theo độ đo nào sẽ phù hợp hơn. Vì vậy, có một độ đo kết hợp cả Precision và Recall là F1. F1 càng cao, mô hình phân lớp càng tốt. Công thức F1 như sau:

A picture containing text

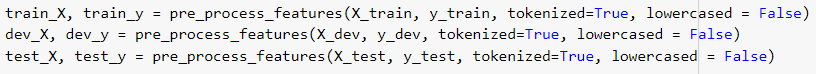
Description automatically generated

*Hình 10: công thức tính F1 score*

# Chương 4: Cài đặt thực nghiệm

## Dữ liệu thực nghiệm

Trong học máy, chúng ta không thể đưa dữ liệu thô trực tiếp vào huấn luyện mô hình. Dữ liệu sau bước tiền xử lý sẽ được vector hóa



*Hình 11: xử lý tiền dữ liệu*

Text

Description automatically generated

*Hình 12: token hóa dữ liệu*

Sau khi đã được vector hóa, dữ liệu lúc này sẽ được đưa vào một dataset được định nghĩa từ trước để phù hợp với thư viện torch – thứ mà mô hình học máy sẽ nhận dữ liệu và bắt đầu quá trình học

Text

Description automatically generated

*Hình 13: định nghĩa dataset mà mô hình sẽ sử dụng*

A screenshot of a computer

Description automatically generated with low confidence

*Hình 14: đưa dữ liệu vào dataset được định nghĩa*

## Phương pháp thực nghiệm

Quá trình thực nghiệm được tiến hành với bộ dữ liệu train, đánh giá bằng dev và được thử nghiệm dự đoán trên test.

Text, letter

Description automatically generated

*Hình 15: huấn luyện mô hình bằng train, đánh giá bằng dev*



*Hình 16: dự đoán label cho test*

## Kết quả thực nghiệm

Sau khi máy thực hiện xong phần huấn luyện, ta thu được một số kết quả như sau:

Text

Description automatically generated

*Hình 17: confusion matrix và hệ số F1 macro, micro*

Calendar

Description automatically generated

*Hình 18: biểu đồ nhiệt confusion matrix*

Table

Description automatically generated

*Hình 19: dataset test với cột dự đoán*

# Chương 5: Demo chương trình

## Tải các thư viện cần thiết



*Hình 20: cài đặt thư viện transformer*

Graphical user interface, text, application

Description automatically generated

*Hình 21: cài đặt thư viện vncorenlp*

Table

Description automatically generated

*Hình 22: tải wordsegmenter*

## Tiền xử lý dữ liệu

Text

Description automatically generated

*Hình 23: đọc dataset*

*Text

Description automatically generated*

*Hình 24: viết hàm tiền xử lý dữ liệu*

Text

Description automatically generated with medium confidence

*Hình 25: tách dataset thành các cột dữ liệu riêng biệt để chuẩn bị bước tiền xử lý*

## Áp dụng mô hình PhoBERT

A picture containing Word

Description automatically generated

*Hình 26: gọi thư viện máy học*

Text

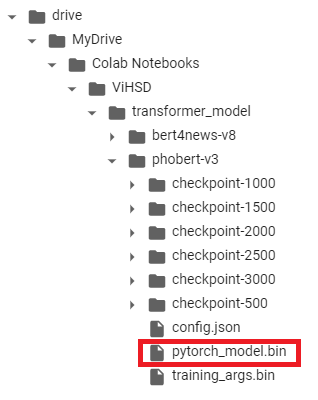
Description automatically generated

*Hình 27: tiền xử lý dữ liệu và đưa vào dataset mà máy sẽ sử dụng*

Text

Description automatically generated

*Hình 28: viết các parameter cần thiết và tiến hành train model, sau đó lưu model*



*Hình 29: nơi chứa model đã được train*

Text

Description automatically generated

*Hình 30: nhận về kết quả dự đoán trên test và hiện các kết quả thu được*

Graphical user interface

Description automatically generated with medium confidence

*Hình 31: kết quả hiện thị quá trình train*

A picture containing text

Description automatically generated

*Hình 32: kết quả khi model chạy trên test và các thông số độ đo cần lưu ý*

Graphical user interface, text, application, chat or text message

Description automatically generated

*Hình 33: vector dự đoán*

Ta có thể thấy rằng sau khi máy trải qua giai đoạn dự đoán t có thể chuyển nó về vector dự đoán với độ dài trùng với độ dài dataset dùng để thử nghiệm (test)

Graphical user interface, text, application

Description automatically generated

*Hình 34: đưa cột dự đoán vào dataset ban đầu để so sánh*

Graphical user interface, application

Description automatically generated with medium confidence

*Hình 35: đối chiếu các kết quả bị dự đoán sai*

## Áp dụng mô hình đã được train cho một bộ dữ liệu chưa có label

Trước khi có thể sử dụng model để dự đoán label của comment, ta phải tiền xử lý dữ liệu và trích xuất vector đặc trưng giống như lúc ta huấn luyện mô hình

Graphical user interface, text, application

Description automatically generated

*Hình 36: tiền xử lý dữ liệu*

Để sử dụng được mô hình đã được huấn luyện, ta phải thực hiện 2 điều: gọi tên của mô hình ta sử dụng (phobert-base như trong hình) và tải các pretrained parameter để load vào mô hình.

A picture containing timeline

Description automatically generated

Sau đó ta chỉ cần lấy kết quả thu được và biến đổi nó thành một vector để gắn vào dataset ban đầu để đối chiếu:

A picture containing table

Description automatically generated

*Hình 37: chuyển dự đoán của máy thành vector*

Text

Description automatically generated

*Hình 38: đưa vào dataset ban đầu để xem kết quả dự đoán của mô hình*

## Áp dụng mô hình Naïve-Bayes

Dữ liệu sau khi được tiền xử lí như mô hình PhoBERT ở trên ta cần phải chuyển hóa về đúng dạng matrix để áp dụng vào mô hình Naive Bayes

Graphical user interface, text, application

Description automatically generated

*Hình 39: Chuyển hóa dữ liệu về dạng matrix và kết quả vector của matrix*

Áp dụng dữ liệu sau khi chuyển hóa vào mô hình Naive Bayes

Graphical user interface, text, application, email

Description automatically generated

*Hình 40: Đưa dữ liệu vào mô hình Naive Bayes*

Chạy kết quả predict cũng như các độ đo cần thiết

Graphical user interface, text, application, email

Description automatically generated

*Hình 41: Chạy kết quả và tính các độ đo cần thiết*

Graphical user interface, application

Description automatically generated

*Hình 42: Kết quả các độ đo cần lưu ý*

# Chương 6: Kết luận

Thông qua quá trình tìm tòi, học hỏi, nghiên cứu, nhóm đã có thể áp dụng được các kỹ thuật trong Khai thác dữ liệu vào một bài toán xử lý dữ liệu truyền thông xã hội. Đầu tiên là đã tiến hành được các bước làm sạch dữ liệu theo từng trường hợp khác nhau. Tiếp theo là áp dụng kỹ thuật rút trích đặc trưng (Feature Extraction) để rút trích vector đặc trưng từ những dòng dữ liệu text. Cuối cùng là huấn luyện mô hình và trực quan hoá được kết quả.

Tuy nhiên, bài báo cáo và quá trình thực nghiệm vẫn còn nhiều thiếu sót. Thứ nhất là do sử dụng tài khoản miễn phí cung cấp bởi Google nên bị giới hạn dung lượng rất nhiều khi xử lý dữ liệu và huấn luyện mô hình, vì vậy các tham số cài đặt cho mô hình không phải là tham số tối ưu mà nhiều bài viết và nhiều người khuyên dùng. Thứ hai là nhóm chưa có nhiều kiến thức chuyên sâu và kinh nghiệm về Deep Learning nên không can thiệp sâu vào quá trình huấn luyện của mô hình BERT (rất nhiều tham số được cài đặt mặc định nên chưa tối ưu). Cuối cùng, quá trình xử lý dữ liệu còn khá cơ bản. Trong tương lai nếu dữ liệu được xử lý cụ thể hơn thì hoàn toàn có thể cải thiện kết quả của mô hình.

# Tài liệu tham khảo

1. <https://github.com/sonlam1102/vihsd> - truy cập lần cuối vào 11/06/2022
2. <https://arxiv.org/pdf/2103.11528.pdf> - truy cập lần cuối vào 11/06/2022
3. <https://github.com/VinAIResearch/PhoBERT> - truy cập lần cuối vào 11/06/2022
4. <https://pytorch.org/tutorials/beginner/saving_loading_models.html> - truy cập lần cuối vào 11/06/2022