ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KHOA HỌC VÀ KỸ THUẬT THÔNG TIN

ĐỒ ÁN CUỐI KỲ

MÔN KHAI THÁC DỮ LIỆU TRUYỀN THÔNG XÃ HỘI

Đề tài 1: PHÂN LOẠI VĂN BẢN TRUYỀN THÔNG XÃ HỘI TRÊN TIẾNG VIỆT

Lớp: IE403.L21 Nhóm: 3

GVHD: ThS. Nguyễn Văn Kiệt

Nhóm sinh viên thực hiện: Nhóm 3

STT	Họ và tên	MSSV
1	Trương Thanh Thiên	18521431
2	Võ Trung Hiếu	18520758
3	Trần Quốc Thành	18521414
4	Trần Anh Thư	18521464
5	Mai Xuân Tú	18521581

TP. Hồ Chí Minh, 6/2021

LỜI CẢM ƠN.

Lời đầu tiên, nhóm 3 xin gửi lời cảm ơn chân thành đến tập thể quý Thầy Cô Trường Đại học Công nghệ thông tin – Đại học Quốc gia TP.HCM và quý Thầy Cô khoa Khoa học và kỹ thuật thông tin đã giúp cho nhóm có những kiến thức cơ bản làm nền tảng để thực hiện đồ án này.

Đặc biệt, nhóm xin gửi lời cảm ơn đến thầy Nguyễn Văn Kiệt (Giảng viên lý thuyết môn Khai thác dữ liệu truyền thông xã hội). Thầy đã cung cấp kiến thức, chỉ bảo trực tiếp, hướng dẫn tận tình, sửa chữa và đóng góp nhiều ý kiến quý báu giúp nhóm hoàn thành tốt đồ án của mình.

Xuất phát từ mục đích học tập, cũng như tìm hiểu thêm về các phương pháp, các bài toán phân loại văn bản, nhóm chúng em đã thực hiện đồ án với đề tài 1: "**Phân loại văn bản truyền thông xã hội trên tiếng Việt**". Trong quá trình thực hiện đồ án dựa trên những kiến thức được Thầy cung cấp trên trường, kết hợp với việc tự tìm hiểu những phương pháp và kiến thức mới, nhóm đã cố gắng thực hiện đồ án một cách tốt nhất. Tuy nhiên, đồ án vẫn còn nhiều sai sót nhưng nó là kết quả của sự nỗ lực của các thành viên trong nhóm, sự giúp đỡ của Thầy.

Nhóm rất mong nhận sự góp ý từ phía Thầy nhằm giúp nhóm rút ra những kinh nghiệm quý báu, hoàn thiện vốn kiến thức đã học tập để là hành trang cho nhóm có thể tiếp tục hoàn thành những đồ án tiếp theo trong tương lai.

Xin chân thành cảm ơn Thầy!

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

……, ngày…….tháng……năm 2021

Người nhận xét

(Ký tên và ghi rõ họ tên)

DANH MỤC CÁC BẢNG, HÌNH ẢNH

Danh mục các Bảng:

Danh mục các Hình ảnh:

MỤC LỤC

Chương 1 GIỚI THIỆU

Những năm gần đây, thuật toán phân loại văn bản nói riêng và lĩnh vực NLP nói chung không ngừng phát triển. Để ứng dụng được các công nghệ mới liên quan đến xử lý ngôn ngữ tự nhiên, đòi hỏi phải tìm hiểu và nắm bắt được cách thức hoạt động của các thuật toán hiện đại để từ đó có hướng xử lý dữ liệu phù hợp.

Phân loại văn bản tuy được nghiên cứu từ rất sớm nhưng đến nay vẫn là một bài toán được giới nghiên cứu NLP quan tâm. Dữ liệu gia tăng nhanh chóng dẫn đến nhu cầu phân tích dữ liệu người dùng tăng cao từ đó xác định được xu hướng, thị hiếu, các vấn đề tiêu biểu và nổi bật tại một thời điểm nào đó.

Bài toán phân loại văn bản (text classification) được thấy rất nhiều trong các ứng dụng NLP (xử lý ngôn ngữ tự nhiên). Mục tiêu của một hệ thống phân loại văn bản là nó có thể tự động phân loại một văn bản cho trước, để xác định xem văn bản đó thuộc thể loại nào. Một số ứng dụng của hệ thống phân loại như:

- Hiểu được ý nghĩa, đánh giá, bình luận của người dùng từ mạng xã hội.
- Phân loại emails là spam hay không spam.
- Tự động gắn thẻ cho những truy vấn, tìm kiếm của người dùng.
- Phân loại các bài báo điện tử.

Bài toán phân loại văn bản là một bài toán học giám sát (supervised learning) trong học máy (machine learning). Bài toán này yêu cầu dữ liệu cần có nhãn (label). Mô hình sẽ học từ dữ liệu có nhãn đó, sau đó được dùng để dự đoán nhãn cho các dữ liệu mới mà mô hình chưa gặp. Để giải quyết một bài toán phân loại văn bản, ta thực hiện 4 bước:

1. Chuẩn bị dữ liệu (Dataset Preparation)
2. Xử lý thuộc tính của dữ liệu (Feature Engineering)
3. Xây dựng mô hình (Build Model)
4. Tinh chỉnh mô hình và cải thiện hiệu năng (Improve Performance)

Phân loại văn bản (Text Classification) là bài toán thuộc nhóm học có giám sát (Supervised learning) trong học máy. Ví dụ bài toán phân loại cảm xúc hay thái độ của

người dùng qua bình luận (comment) trên các trang phim, MV ca nhạc, đánh giá về sản phẩm …, trong đó có một bài toàn khá là nổi tiếng. Đó là phân loại cảm xúc (Sentiment Analysis). Một số bài toán khác như là: phân loại văn bản dựa theo chủ đề, topic, ...

Chương 2 CÁC CÔNG TRÌNH LIÊN QUAN

2.1 Các công trình liên quan đến Bộ dữ liệu

Trong đồ án này, nhóm sử dụng 3 bộ dữ liệu đã được công bố tại các hội nghị lớn:

- Bộ dữ liệu UIT-VSMEC [1]: Emotion Recognition for Vietnamese Social Media Text (*Pacific Association for Computational Linguistics*) của nhóm tác giả đến từ trường Đại học Công nghệ Thông tin và Đại học Khoa học xã hội và Nhân văn – Đại học Quốc gia Thành phố Hồ Chí Minh: Vong Anh Ho, Duong Huynh-Cong Nguyen, Danh Hoang Nguyen, Linh Thi-Van Pham, Duc-Vu Nguyen, Kiet Van Nguyen, Ngan Luu-Thuy Nguyen. Bài báo được đăng tải tại hội nghị Springer, Singapore vào năm 2019.
- Bộ dữ liệu UIT-VSFC [2]: Vietnamese Students' Feedback Corpus for Sentiment Analysis của nhóm tác giả đến từ trường Đại học Công nghệ Thông tin – Đại học Quốc gia Thành phố Hồ Chí Minh: Kiet Van Nguyen; Vu Duc Nguyen; Phu X. V. Nguyen; Tham T. H. Truong; Ngan Luu-Thuy Nguyen . Bài báo được đăng tại *2018 10th International Conference on Knowledge and Systems Engineering (KSE)*, IEEE vào năm 2018.
- Bộ dữ liệu UIT-ViCTSD [3]: Constructive and Toxic Speech Detection for Open-domain Social Media Comments in Vietnamese của nhóm tác giả đến từ trường Đại học Công nghệ Thông tin – Đại học Quốc gia Thành phố Hồ Chí Minh: Luan Thanh Nguyen, Kiet Van Nguyen, Ngan Luu-Thuy Nguyen. Bài báo được chấp nhận đăng tải tại *The 34th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA/AIE 2021).*

2.2 Các công trình liên quan đến Phương pháp

Giới hạn trong nội dung của đồ án này, nhóm sử dụng 5 phương pháp với các công trình liên quan đến các phương pháp như sau:

- Logistic Regression
- CNN-LSTM nhóm có tham khảo bài báo *Class-based LSTM Russian language model with linguistic information* [4] được đăng tải tại Proceedings of The 12th

Language Resources and Evaluation Conference vào năm 2020 của nhóm tác giả Irina Kipyatkova, Alexey Karpov.

- Với các phương pháp SOTA (PhoBert, Bert4News, XLM-R) nhóm có tham khảo công trình *Investigating Monolingual and Multilingual BERTModels for Vietnamese Aspect Category Detection* [5] của nhóm tác giả: Dang Van Thin, Lac Si Le, Vu Xuan Hoang, Ngan Luu-Thuy Nguyen đến từ trường Đại học Công nghệ Thông tin – Đại học Quốc gia Thành phố Hồ Chí Minh.
- Ngoài ra riêng phương pháp PhoBert, nhóm có tham khảo bài báo trình bày về Roberta trong bài báo *Roberta: A robustly optimized bert pretraining approach* [6] của nhóm tác giả: Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, Veselin Stoyanov với 1377 lượt cite và bài báo về PhoBert của nhóm tác giả Dat Quoc Nguyen, Anh Tuan Nguyen: *PhoBERT: Pre-trained language models for Vietnamese* [7] với 51 lượt cite.
- Đối với Bert4News nhóm có tham khảo bài báo: *Compose transformer pretrained models for Reliable Intelligence Identification on Social network* [8] của nhóm tác giả Thanh Chinh Nguyen, Van Nha Nguyen.
- Với phương pháp XLM-R nhóm cáo tham khảo công trình liên quan về XLM với 642 lượt cite: *Cross-lingual language model pretraining* [9] của nhóm tác giả Guillaume Lample, Alexis Conneau. Và *Unsupervised cross-lingual representation learning at scale* [10] - bài báo liên quan đến XLM-R của nhóm tác giả Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, Veselin Stoyanov với 697 lượt cite.

Chương 3 CÁC BỘ DỮ LIỆU SỬ DỤNG

Trong đồ án môn học này, chúng tôi thực nghiệm trên 3 bộ ngữ liệu được quy định sẵn cho đề tài số 1 – Phân loại văn bản. Các bộ ngữ liệu đều là ngữ liệu tiếng Việt, được công bố và sử dụng tại các bài báo và công trình nghiên cứu uy tín [1] [2] [3].

3.1 UIT-VSMEC

– Bộ dữ liệu UIT-VSMEC [1] được cung cấp thông qua bài báo tại: https://arxiv.org/abs/1911.09339.

– Bộ ngữ liệu là dữ liệu bình luận của người dùng trên mạng xã hội đã được thu thập và gán nhãn. Trong bộ ngữ liệu này, biến phân loại là Emotion với miền giá trị là 1 trong số 7 giá trị sau: ‘Other’, ‘Disgust’, ‘Enjoyment’, ‘Anger’, ‘Surprise’, ‘Sadness’, ‘Fear’.

– Bộ ngữ liệu được các tác giả chia thành 3 tập Train, Dev, Test với kích thước từng tập:

+ Tập Train: 5548 dòng dữ liệu.

+ Tập Dev: 686 dòng dữ liệu.

+ Tập Test: 693 dòng dữ liệu.

– Bộ dữ liệu mang tính thách thức từ nội dung đến hình thức. Bản chất là câu bình luận trên Mạng xã hội nên ngữ liệu có tính phi cấu trúc cao, câu từ và ngữ pháp tự do. Nhiều câu bình luận sai chính tả và có kèm emotion icon, chính 2 tính chất này đã góp phần tăng độ thách thức cho bộ ngữ liệu.

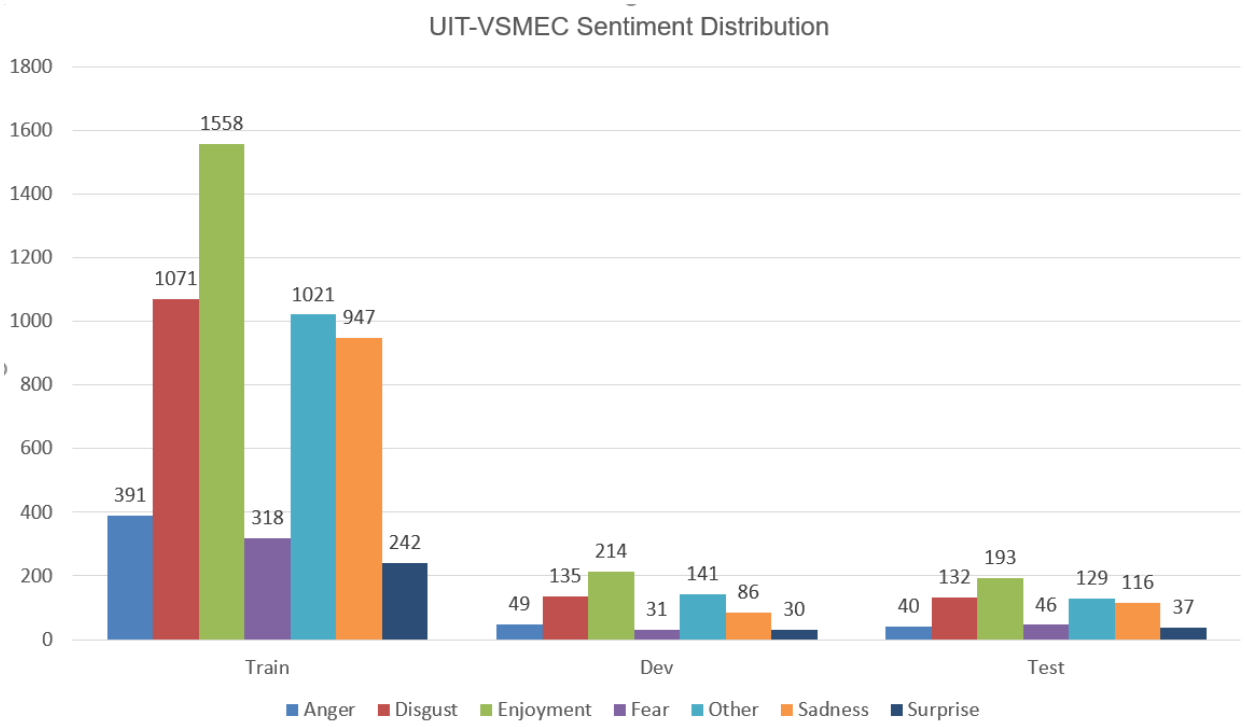
	Unnamed: 0	Emotion	Sentence
126	1396	Disgust	cô tiên béo quá .. 😅😅 giọng còn the thé nữa .. 😅😅
110	1099	Sadness	thế làm sao dỗ bạn gái đang giận :(
136	114	Disgust	thằng cha bảo vệ có tâm quá . nge con nhỏ kêu ...
170	1420	Anger	immm
172	1044	Enjoyment	đa´ người yêuc quá´

Hình 3.1: Dữ liệu mẫu cho bộ ngữ liệu UIT-VSMEC

– Chúng tôi tiến hành phân tích phân phối giá trị nhãn của bộ ngữ liệu này và rút ra được 1 số nhận xét:

+ Phân bố giá trị nhãn của biến Emotion không đồng đều và có số lượng chênh lệch cao giữa các nhãn.

+ Ở tập Train, nhãn có số lượng thấp nhất là 242 (‘Surprise’) trong khi nhãn nhiều nhất là 1558 (‘Enjoyment’).



Hình 3.2: Phân bố tập giá trị nhãn biến Emotion của bộ ngữ liệu UIT-VSMEC

3.2 UIT-VSFC

– Bộ dữ liệu UIT-VSFC [2] được cung cấp thông qua bài báo tại:

https://www.academia.edu/40956009/UIT-VSFC_Vietnamese_Students_Feedback_Corpus_for_Sentiment_Analysis.

– Bộ ngữ liệu là dữ liệu đánh giá sau môn học của sinh viên với 2 biến phân loại là Sentiments và Topic với miền giá trị lần lượt là {0, 1, 2} cho Sentiments và {0, 1, 2, 3} cho Topic.

– Bộ ngữ liệu được các tác giả chia thành 3 tập Train, Dev, Test với kích thước từng tập:

+ Tập Train: 11426 dòng dữ liệu
+ Tập Dev: 1583 dòng dữ liệu
+ Tập Test: 3166 dòng dữ liệu

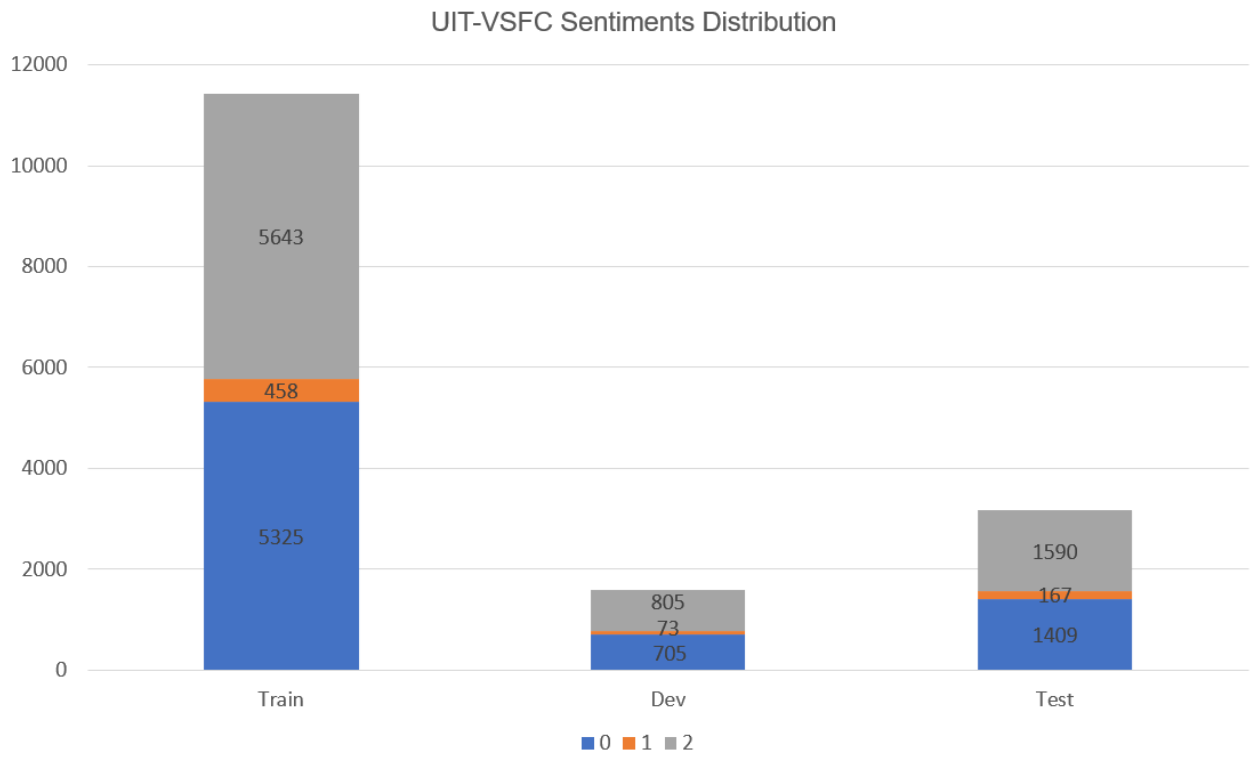
– So với UIT-VSMEC, bộ ngữ liệu này có phần ít thách thức hơn, do bản chất là lời nhận xét và đánh giá của sinh viên sau môn học nên câu từ có tính câu trúc hơn, hiếm có những câu sai chính tả hoặc câu có chứa emotion icon.

	Sents	Sentiments	Topic
11146	giảng viên cần hỗ trợ sinh viên nhiều hơn , hư...	0	0
2748	nhiều thử thách cho sinh viên .	0	1
4485	thầy rất thường xuyên kể về các câu chuyện cá ...	0	0
9618	chưa giải quyết thắc mắc của sinh viên thỏa đá...	0	0
5676	giảng viên trên lớp liên hệ thực tế quá nhiều .	0	0

Hình 3.3: Dữ liệu mẫu cho bộ ngữ liệu UIT-VSFC

– Chúng tôi tiến hành phân tích phân phối giá trị nhãn của bộ ngữ liệu này và rút ra được một số nhận xét:

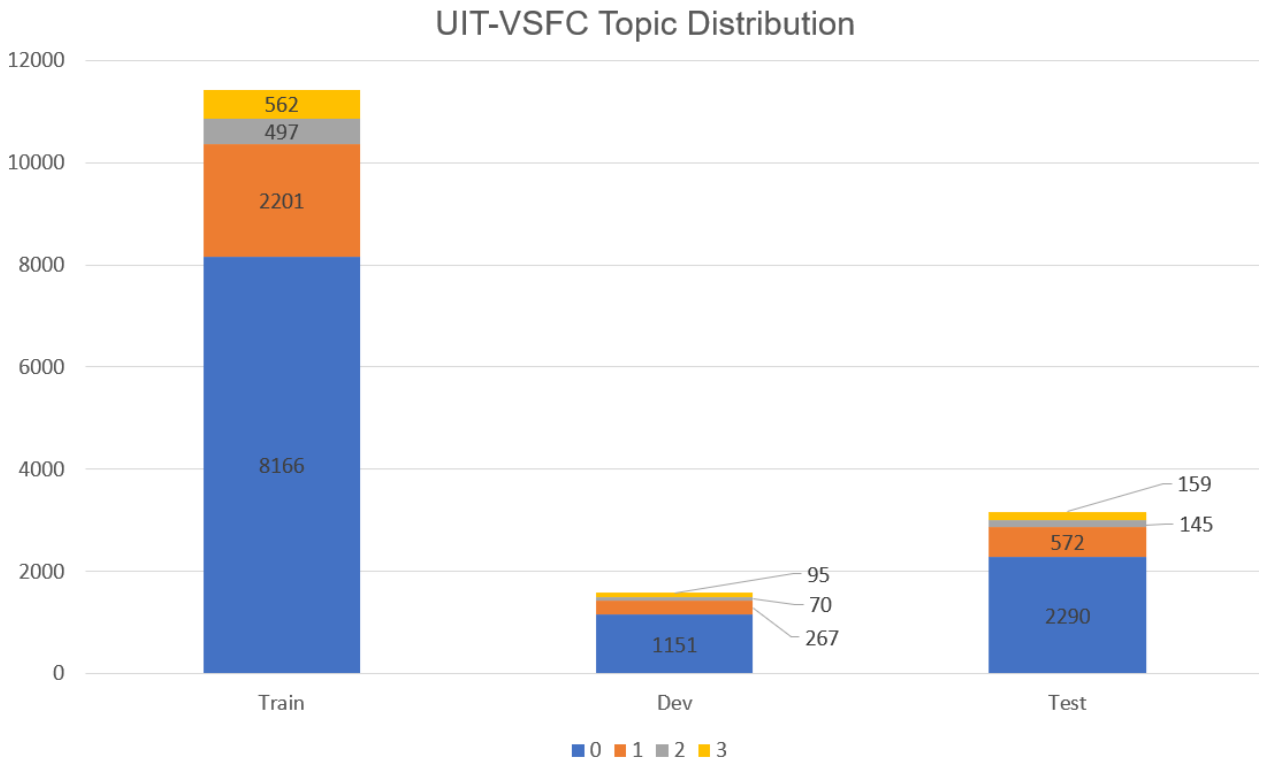
+ Đổi với biến **Sentiments**:
  - Phân bố giá trị nhãn của biến Sentiments không đồng đều và có số lượng chênh lệch cao giữa các nhãn.
  - Ở tập Train, nhãn có số lượng thấp nhất là 458 (‘1’) trong khi nhãn nhiều nhất là 5643 (‘2’).



Hình 3.4: Phân bố tập giá trị nhãn biến Sentiments của bộ ngữ liệu UIT-VSFC

+ Đối với biến **Topic**:

  - Phân bố giá trị nhãn của biến Topic không đồng đều và có số lượng chênh lệch cao giữa các nhãn.
  - Ở tập Train, nhãn có số lượng thấp nhất là 497 (‘2’) trong khi nhãn nhiều nhất là 8166 (‘0’).



Hình 3.5: Phân bố tập giá trị nhãn biến Topic của bộ ngữ liệu UIT-VSFC

3.3 UIT-ViCTSD

– Bộ dữ liệu UIT-ViCTSD [3] được cung cấp thông qua bài báo tại:

https://arxiv.org/abs/2103.10069.

– Bộ ngữ liệu là dữ liệu bình luận của người dùng cho 1 bài viết theo từng chủ đề cụ thể. Bộ ngữ liệu này bao gồm 2 biến phân loại là Constructiveness và Toxicity với cùng miền giá trị là {0, 1}.

– Bộ ngữ liệu được các tác giả chia thành 3 tập Train, Dev, Test với kích thước từng tập:

+ Tập Train: 7000 dòng dữ liệu.

+ Tập Dev: 2000 dòng dữ liệu.

+ Tập Test: 1000 dòng dữ liệu.

– UIT-ViCTSD tương đồng với UIT-VSMEC, bộ ngữ liệu này cũng là dữ liệu bình luận của người dùng trên mạng xã hội, tuy nhiên các bình luận này được thu thập theo từng chủ đề cụ thể.

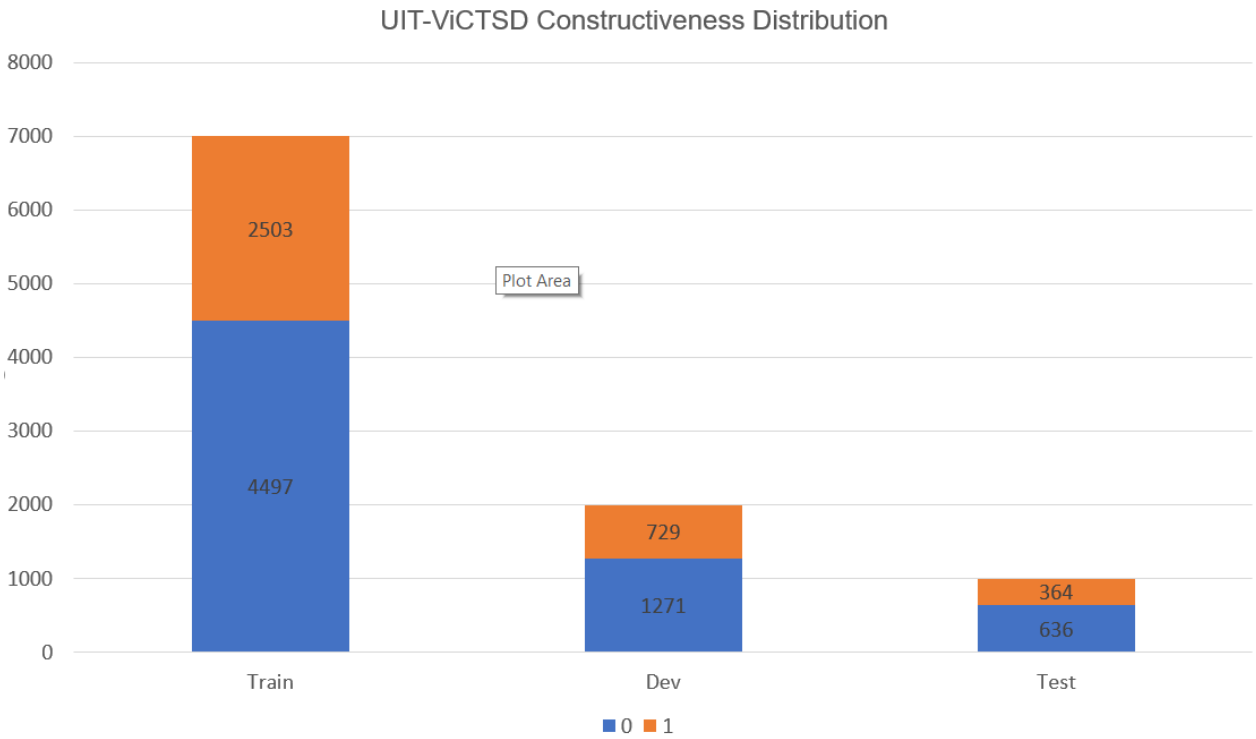
	Unnamed: 0	Comment	Constructiveness	Toxicity	Title	Topic
6584	3158	Đúng là bán những cỗ máy bay trên trời thì giá...	0	0	A380 - Siêu máy bay thất sủng của Airbus	KinhDoanh
2465	6597	Đâu phải ai cũng có thời gian để cả ngày theo ...	0	0	Ba người Quảng Nam ngộ độc sau ăn pate Minh Chay	SucKhoe
2951	6059	mong chúng ta mãi bình yên.	0	0	14 ngày không lây nhiễm nCoV cộng đồng	SucKhoe
324	2409	Loài cá sấu này sinh sống ở vùng đất ngập nước...	1	0	Cá sấu chồm lên lôi cổ chó nhà xuống nước	KhoaHoc
2131	3768	Đã gọi là bất động thì chả ai biết trước được ...	1	0	Có nên rút 3 tỷ đang gửi ngân hàng để mua đất ...	KinhDoanh

Hình 3.6: Dữ liệu mẫu cho bộ ngữ liệu UIT-ViCTSD

– Chúng tôi tiến hành phân tích phân phối giá trị nhãn của bộ ngữ liệu này và rút ra được 1 số nhận xét:

+ Đối với biến **Sentiments**:

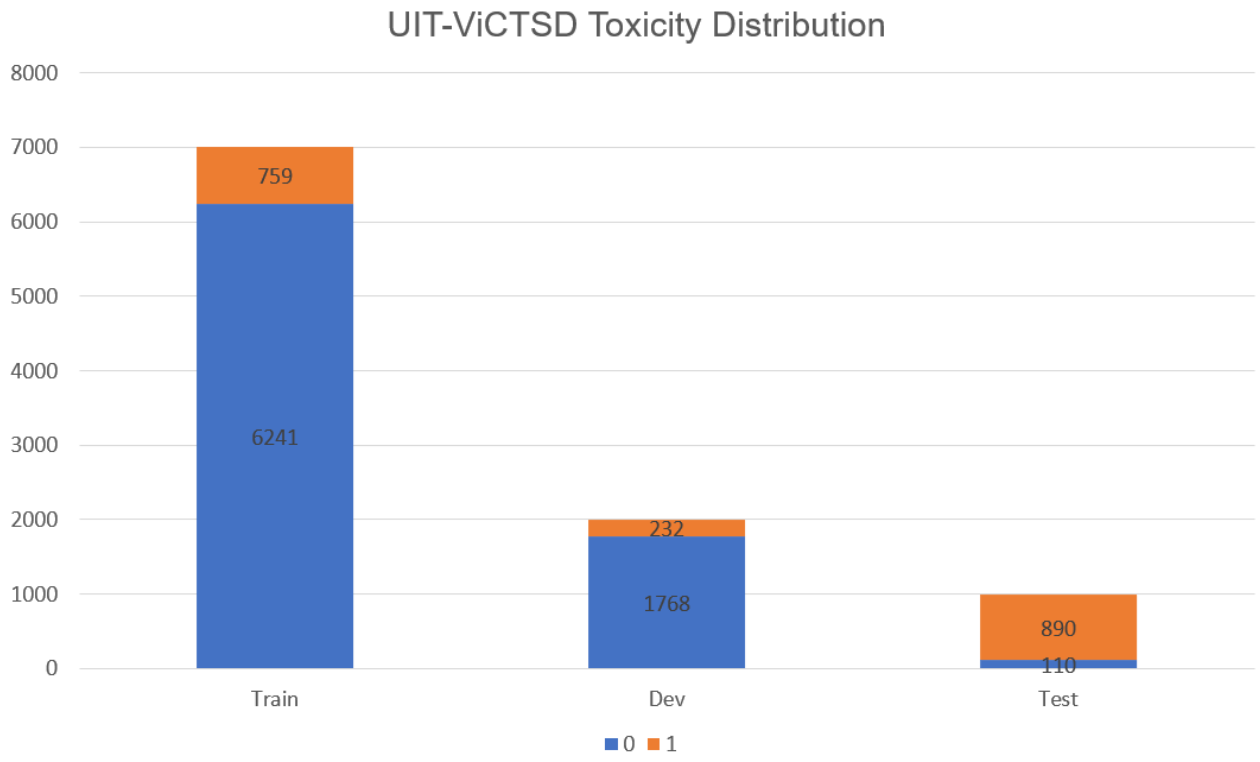
- Phân bố giá trị nhãn của biến Constructiveness không đồng đều và có số lượng chênh lệch khá cao giữa các nhãn.
- Ở tập Train, nhãn có số lượng thấp nhất là 2503 (‘1’) trong khi nhãn nhiều nhất là 4497 (‘0’). Tuy nhiên, xét về phân bố thì biến Constructiveness vẫn cân bằng hơn so với biến Toxicity.



Hình 3.7: Phân bố tập giá trị nhãn biến Constructiveness của bộ ngữ liệu UIT-ViCTSD

+ Đối với biến **Toxicity**:

  - Phân bố giá trị nhãn của biến Toxicity không đồng đều và có số lượng chênh lệch rất cao giữa các nhãn.
  - Ở tập Train, nhãn có số lượng thấp nhất là 759 (‘1’) trong khi nhãn nhiều nhất là 6241 (‘0’).



Hình 3.8: Phân bố tập giá trị nhãn biến Toxicity của bộ ngữ liệu UIT- ViCTSD

Chương 4 CÁC PHƯƠNG PHÁP THỰC NGHIỆM

Trong phạm vi và yêu cầu của đề tài môn học này, chúng tôi tiến hành thực nghiệm trên 5 phương pháp sau khi đã chọn lọc từ nhiều phương pháp khác nhau. 5 phương pháp được lựa chọn thuộc về 3 nhóm phương pháp:

– Nhóm phương pháp học máy truyền thống: Logistic Regression

– Nhóm phương pháp deep learning có sử dụng kiến trúc mạng tích chập CNN: CNN-LSTM

– Nhóm phương pháp SOTA dựa trên Transfer Learning:

+ PhoBERT

+ BERT4NEWS

+ XLM-R

4.1 Logistic Regression

4.1.1 Logistic Regression:

– Logistic Regression được biến đổi một chút từ Linear Regression (hồi quy tuyến tính đa biến), bằng cách cho kết quả của Linear Regression vào hàm $sigmoid$, cụ thể:

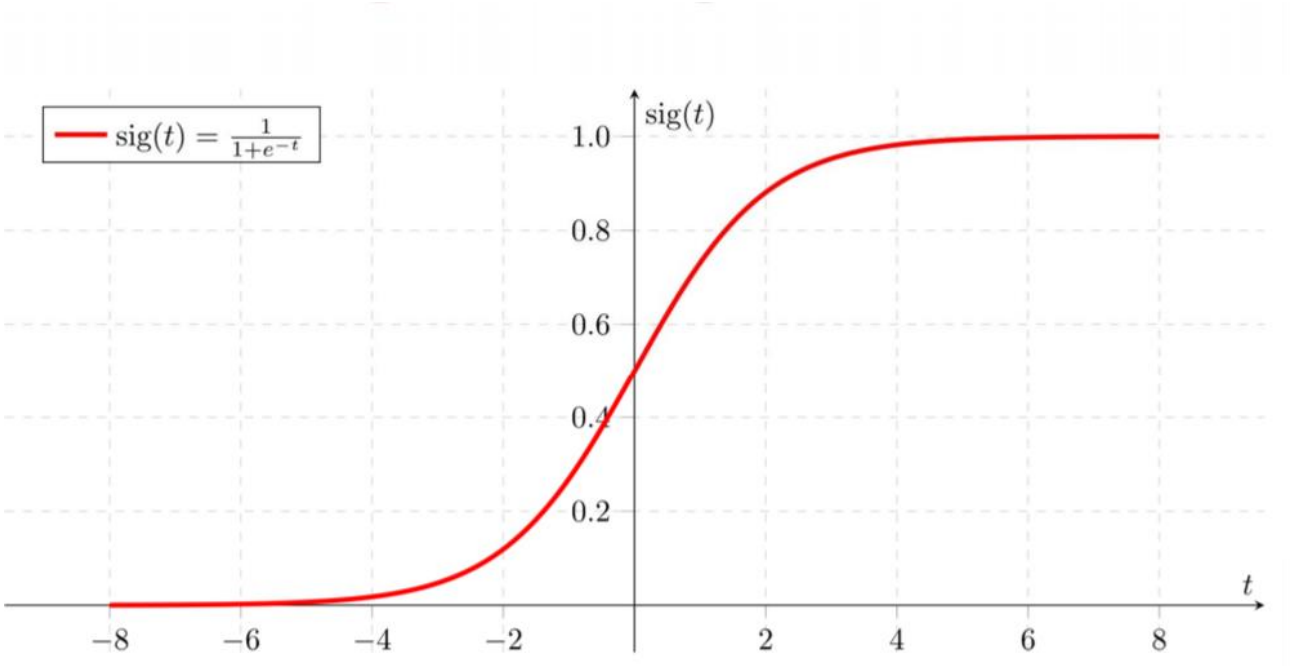
$$p(y_0|x) = sigmoid(a) = \frac{1}{1 + \exp(-a)} = \sigma(w^T x)$$

Trong đó:

+ $p(y_0|x)$ là xác xuất thuộc lớp y_0 với đầu vào x.

+ x là thuộc tính đầu vào, w là trọng số tương ứng.

– Đồ thị hàm sigmoid có dạng:



Hình 4.1: Đồ thị hàm sigmoid

+ Có công thức tính được xác suất rồi thì ta có thể sử dụng một ngưỡng $\epsilon \in [0,1]$ để quyết định nhóm tương ứng. Cụ thể:

$$\begin{cases} x \in y_0 & if\ p(y_0|x) \geq \epsilon \\ x \in y_1 & if\ p(y_0|x) < \epsilon \end{cases}$$

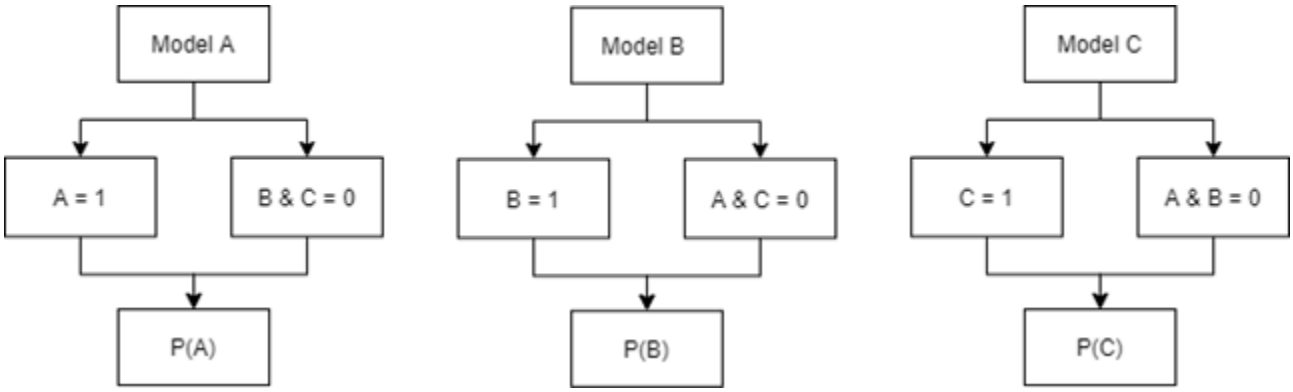
+ Ví dụ: $\epsilon = 0.7$ thì $x \in y_0$ khi mà xác suất thuộc nhóm y_0 của nó là trên 70%, còn dưới 70% thì ta phân nó vào nhóm y_1.

4.1.2 Multinomial Logistic Regression

– Multinominal Logistic Regression: mở rộng thuật toán logistic regression trên nhiều lớp.

– Về cơ bản 2 có 2 phương pháp chính là:

+ Chia K models cho K lớp.



Hình 4.2: Ba models cho 3 lớp

+ Dựa trên mô hình xác xuất nhiều nhóm.

  Sử dụng hàm softmax để tính xác xuất:

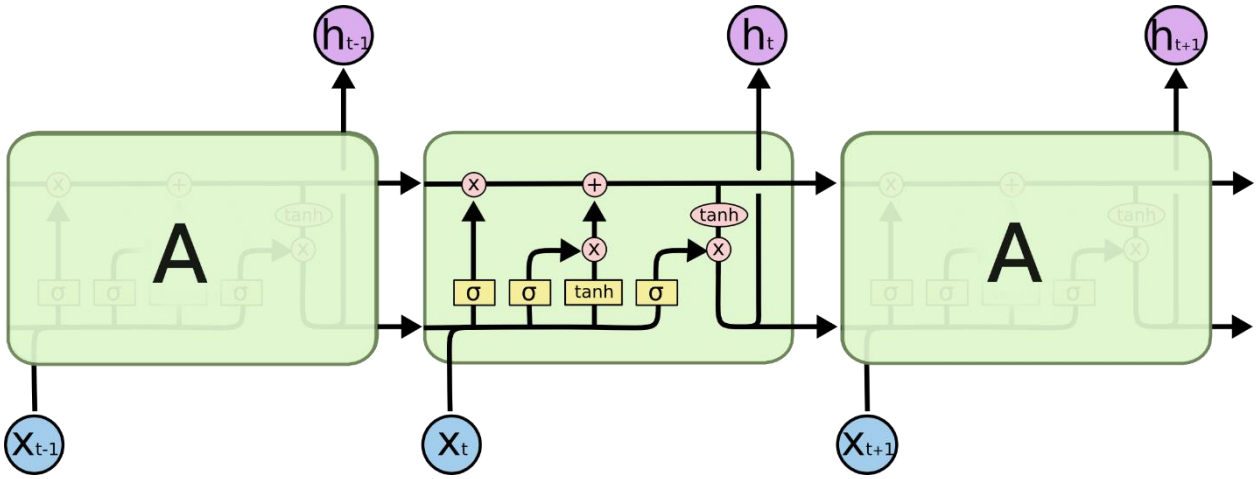
$$p(y_k|x) = p_k = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

  Trong đó: $a_j = \boldsymbol{w}_j^T \boldsymbol{x}$, trong đó véc-tơ $\boldsymbol{w}_j$ là trọng số tương ứng với mỗi nhóm.

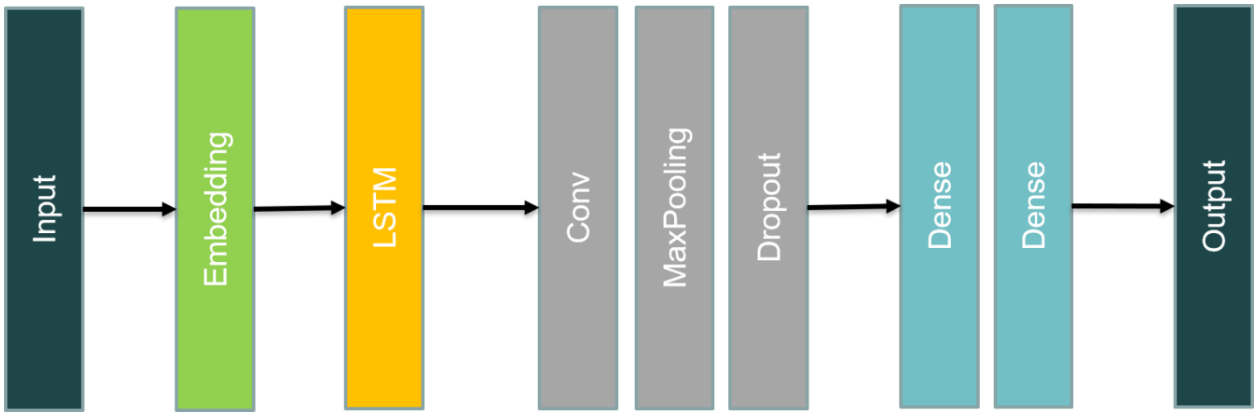
4.2 CNN-LSTM

– LSTM [4] là kiến trúc mạng học sâu được phát triển từ RNN. Về cơ bản, mô hình này khác phục được nhược điểm lớn nhất của RNN đó là việc làm mất đi thông tin ở những từ đứng trước mang ý nghĩa ngữ cảnh hoặc chứa thông tin quan trọng sau mỗi lần lặp. LSTM với kiến trúc phức tạp hơn, cho phép nó giữ lại và truyền thông tin ngữ cảnh ở những từ phía trước đến các node ở sau bằng các connection.

– CNN-LSTM là mạng học sâu kết hợp giữa bộ trích xuất đặc trưng CNN và mô hình LSTM với mong muốn đạt được độ chính xác cao trên các bộ dữ liệu phức tạp.



Hình 4.3: Mô hình LSTM

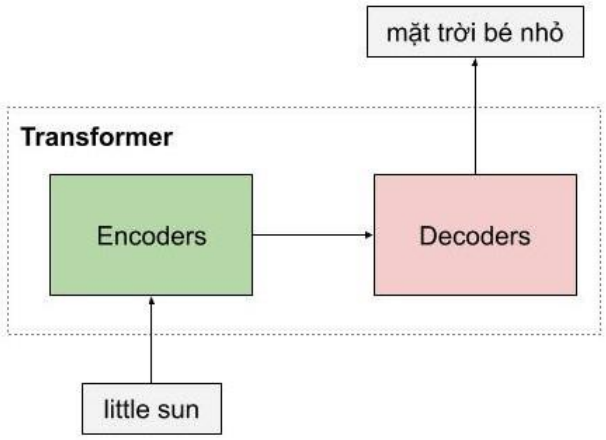


Hình 4.4: Kiến trúc CNN-LSTM được sử dụng

4.3 PhoBert

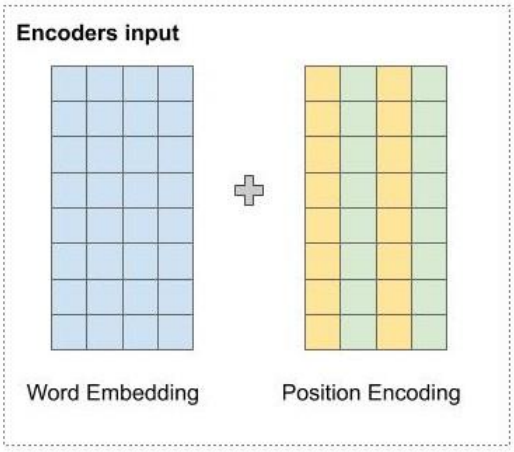
4.3.1 Tìm hiểu về Transformer

– Kiến trúc tổng quan của mô hình transformer bao gồm 2 phần lớn là encoder và decoder. Encoder dùng để học vector biểu của câu với mong muốn rằng vector này mang thông tin hoàn hảo của câu đó. Decoder thực hiện chức năng chuyển vector biểu diễn kia thành ngôn ngữ đích.



Hình 4.5: Mô tả tổng quát hoạt động của Transformer

- Một trong những ưu điểm của transformer là mô hình này có khả năng xử lý song song cho các từ. Encoders của mô hình transformer là một dạng feedforward neural nets, bao gồm nhiều encoder layer khác, mỗi encoder layer này xử lý đồng thời các từ. Trong khi đó, với mô hình LSTM, thì các từ phải được xử lý tuần tự. Ngoài ra, mô hình transformer còn xử lý câu đầu vào theo 2 hướng mà không cần phải stack thêm một LSTM nữa như trong kiến trúc Bidirectional LSTM.
- Cơ chế Position Encoding: dùng để đưa thông tin về vị trí của các từ vào mô hình transformer. Vị trí của các từ được mã hoá bằng một vector có kích thước bằng word embedding và được cộng trực tiếp vào word embedding.



Hình 4.6: Cơ chế Position Encoding

– Cụ thể, tại vị trí chẵn sử dụng hàm sin và hàm cos tại vị trí lẻ:

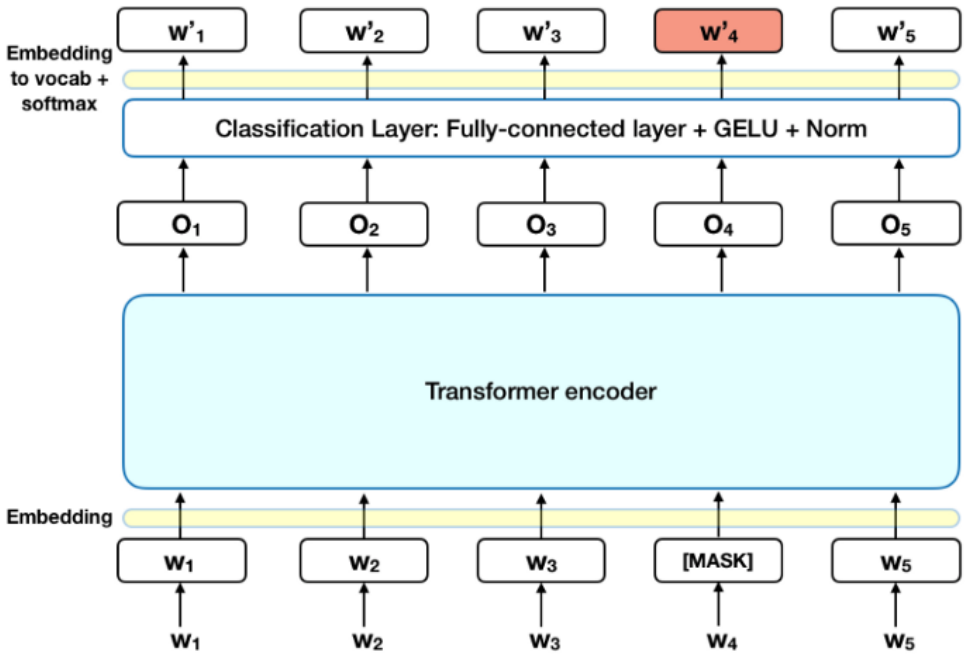
$$p_t^i = f(t)^i = \begin{cases} \sin(\omega_k * t) \ với\ i = 2k \\ \cos(\omega_k * t)\ với\ i = 2k+1 \end{cases} trong\ đó\ \omega_k = \frac{1}{10000^{\frac{2k}{d}}}$$

– Cơ chế Self Attention: cho phép mô hình khi mã hoá một từ có thể sử dụng thông tin của những từ liên quan tới nó.

4.3.2 BERT

– BERT [5] – Bidirectional Encoder Representations from Transformers là một mô hình ngôn ngữ (Language Model) được tạo ra bởi Google AI. BERT được coi là một đột phá lớn trong Machine Learning bởi vì khả năng ứng dụng của nó vào nhiều bài toán NLP khác nhau: Question Answering, Natural Language Inference, … với kết quả rất tốt.

– BERT sử dụng Transformer là một mô hình attention học mối tương quan giữa các từ (hoặc 1 phần của từ) trong một văn bản. Transformer bao gồm Encoder – dùng để đọc dữ liệu đầu vào và Decoder dùng để đưa ra dự đoán. Ở đây, BERT chỉ sử dụng Encoder.



Hình 4.7: Sơ đồ kiến trúc của BERT

– Hình trên mô tả nguyên lý hoạt động của Encoder. Theo đó, input đầu vào là một chuỗi các token w_1, w_2, ... được biểu diễn thành chuỗi các vector trước khi đưa vào

trong mạng neural. Output của mô hình là chuỗi các vector có kích thước đúng bằng kích thước input. Ngoài ra, BERT sử dụng 2 chiến lược training như sau:

+ Mask LM (MLM): Trước khi đưa vào BERT, thì 15% số từ trong chuỗi được thay thế bởi token **MASK**, khi đó mô hình sẽ dự đoán từ được thay thế bởi **MASK** với context là các từ không bị thay thế bởi **MASK**. Hàm lỗi của BERT chỉ tập trung vào đánh giá các từ được đánh dấu **MASK** mà bỏ qua những từ còn lại, do đó mô hình hội tụ chậm hơn so với các mô hình directionl, nhưng chính điều này giúp cho mô hình hiểu ngữ cảnh tốt hơn.

+ NSP – Next Sentence Prediction: Trong chiến lược này, thì mô hình sử dụng một cặp câu là dữ liệu đầu vào và dự đoán câu thứ 2 là câu tiếp theo của câu thứ 1 hay không. Trong quá trình huấn luyện, 50% lượng dữ liệu đầu vào là cặp câu trong đó câu thứ 2 thực sự là câu tiếp theo của câu thứ 1, 50% còn lại thì câu thứ 2 được chọn ngẫu nhiên từ tập dữ liệu.

– Tuỳ vào bài toán mà ta có các phương pháp fine – tune khác nhau:

+ Đối với bài toán Classification, ta thêm một Classification Layer với input là output của Transformer.

+ Đối với bài toàn Question Answering, model nhận dữ liệu input là đoạn văn bản cùng câu hỏi và được huấn luyện để đánh nhãn cho câu trả lời trong đoạn văn bản đó.

+ Đối với bài toán NER (Name Entity Recognition), model được dự đoán nhãn cho mỗi token (tên người, tổ chức, địa danh, …).

4.3.3 RoBERTa

– RoBERTa [6] được xây dựng trên chiến thuật MLM của BERT. RoBERTa có một số cải tiến được thêm vào kiến trúc BERT ban đầu:

+ Mô hình hoá môn ngữ có **MASK** (MLM) được thực hiện masking động thay vì masking tĩnh.

+ Mục tiêu training NSP (dự đoán câu kế tiếp) bị loại bỏ hoàn toàn.

+ Các bước tối ưu hoá 500K được thực hiện trên các lô nhỏ có kích thước 8000 thay vì 1000K bước trên các lô nhỏ có kích thước 256.
+ Mã hoá văn bản được xử lý bằng cách triển khai BPE sử dụng byte làm khối xây dựng thay vì ký tự Unicode.
+ Thử nghiệm trên nhiều dữ liệu hơn (từ 16GB đến 160GB).

4.3.4 PhoBERT

– PhoBert [7] là một pre – trained được huấn luyện monolingual language, tức là chỉ huấn luyện dành riêng cho tiếng Việt. Việc huấn luyện dựa trên kiến trúc và cách tiếp cận giống RoBERTa.

– Tương tự như BERT: PhoBERT có 2 phiên bản là PhoBERT base (12 transformer block) và PhoBERT large (24 transformer block).

– PhoBERT được train trên khoảng 20GB dữ liệu bao gồm khoảng 1GB Vietnamese Wikipedia Corpus và 19GB lấy từ Vietnamese News Corpus. PhoBERT sử dụng RDSegmenter của VNCoreNLP để tách từ cho dữ liệu đầu vào trước khi qua BPE Encode.

4.4 Bert4News

– Bert4News [5] [8] là mô hình Monolingual Pretrained transformer (được nhắc đến tại mục 4.3.1) cho tiếng Việt.

– Bert4News tinh chỉnh dựa trên kiến trúc gốc của Bert (được nhắc đến tại mục 4.3.2) trên bộ dataset tin tức tiếng Việt 20 GB (đã được tách từ ở mức âm tiết).

– Mô hình đạt 0.90268 trên public leaderboard (score chiến thắng là 0.90087).

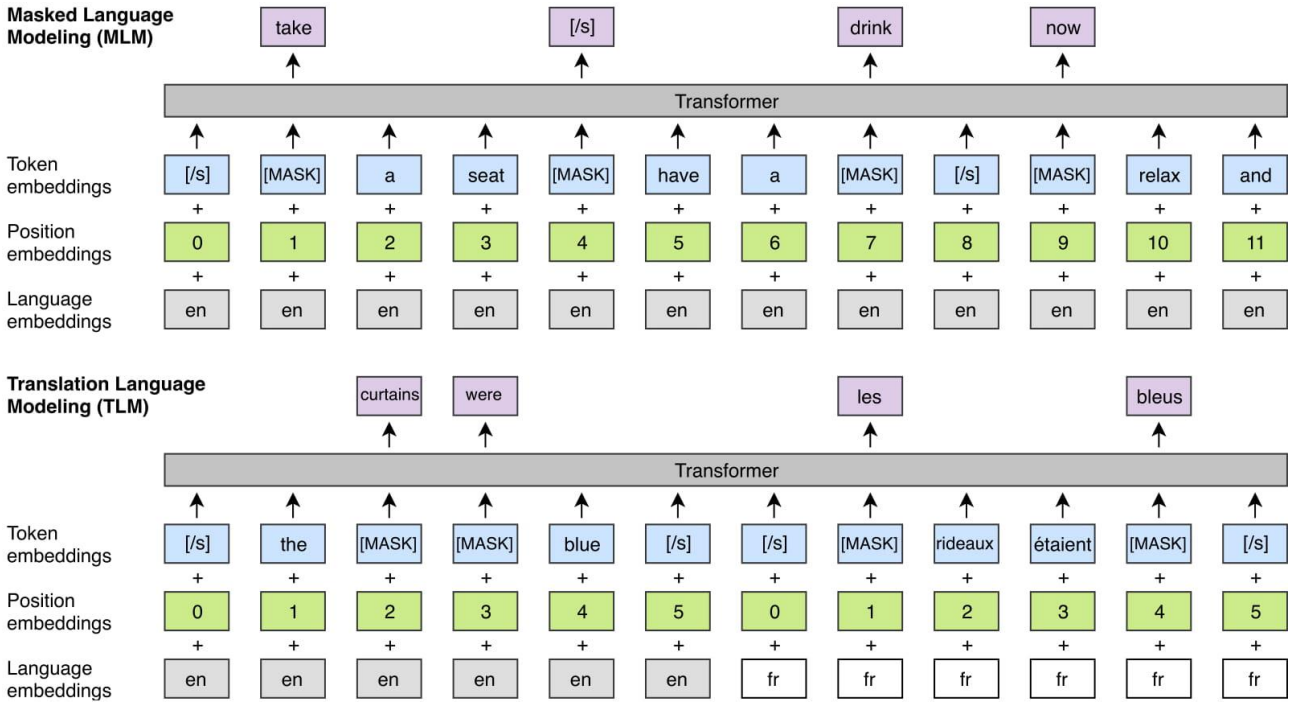
– Bert4News sử dụng word sentencepiece, basic bert tokenization và giống cách cầu hình bert với lowercase = False.

– Mô hình này được sử dụng để tinh chỉnh các vấn đề liên quan đến NLP trên tiếng Việt như Phân đoạn từ (Word Segmentation), Nhận dạng đối tượng được đặt tên, (Named Entity Recognition).

4.5 XLM-R

4.5.1 XLM

- XLM [9] dựa trên Transformer (được nhắc đến tại mục 4.3.1), giống như BERT (được nhắc đến tại mục 4.3.2), được train với Masked Language Modeling (MLM) (được nhắc đến tại mục 4.3.2).
- Ngoài ra, XLM được train với Translation Language Modeling (TLM) nhằm cố gắng buộc mô hình học các cách biểu diễn tương tự cho các ngôn ngữ khác nhau.
- TLM khá đơn giản: nhập cùng một câu bằng hai ngôn ngữ khác nhau và mask tokens như bình thường. Để dự đoán mask tokens, mô hình sau đó có thể chọn sử dụng mã thông báo từ ngôn ngữ khác.
- TLM yêu cầu một tập dữ liệu gồm các câu song song, điều này có thể gây khó khăn.



Hình 4.8: Hình minh họa MLM và TLM trong XLM [9]

4.5.2 XLM-R

- XLM-R [10]: là viết tắt của XLM-RoBERTa (được nhắc đến tại mục 4.3.3). Dựa trên tên, sẽ tự nhiên giả định rằng đó là XLM với RoBERTa thay vì BERT, nhưng điều đó là sai.

- Về cơ bản, nó là một bản cập nhật cho model XLM-100 của Facebook.
- Được train theo kiểu RoBERTa, tức là chỉ sử dụng MLM, không sử dụng TLM và Next Sentence Prediction (NSP).
- XLM-R hoạt động đặc biệt tốt trên các ngôn ngữ ít tài nguyên.
- **Những thay đổi đáng chú ý trong mô hình XLM-R là:**
  + **Data**: XLM-R được train dựa trên dữ liệu CommonCrawl được làm sạch (2.5TB), lớn hơn so với kho dữ liệu Wiki-100 được sử dụng để train các mô hình đa ngôn ngữ khác.
  + **Vocabulary**: XLM-R vocabulary chứa 250 nghìn tokens trái ngược với RoBERTa có 50 nghìn tokens trong vocabulary của nó. Nó sử dụng một Sentence Piece Model (SPM) được chia sẻ rộng lớn để tokenize (mã hóa) các từ của tất cả các ngôn ngữ thay vì sử dụng các tokenizer khác nhau cho các ngôn ngữ khác nhau như XLM-100 model.
  + **XLM-R là mô hình tự giám sát (self-supervised)**, trong khi XLM-100 là mô hình được giám sát (supervised). XLM-R [10] lấy mẫu văn bản từ mỗi ngôn ngữ và đào tạo mô hình để dự đoán masked tokens. XLM-100 yêu cầu các câu song song (các câu có cùng nghĩa) bằng hai ngôn ngữ khác nhau làm đầu vào.
- **XLM-R khác với RoBERTa:**

	Tokens	Parameters
XLM-R	250 nghìn	550 triệu
RoBERTa	50 nghìn	355 triệu

Bảng 1: Bảng so sánh số lượng Tokens và Parameters giữa XLM-R và RoBERTa

Chương 5 THỰC NGHIỆM

5.1 Thực nghiệm

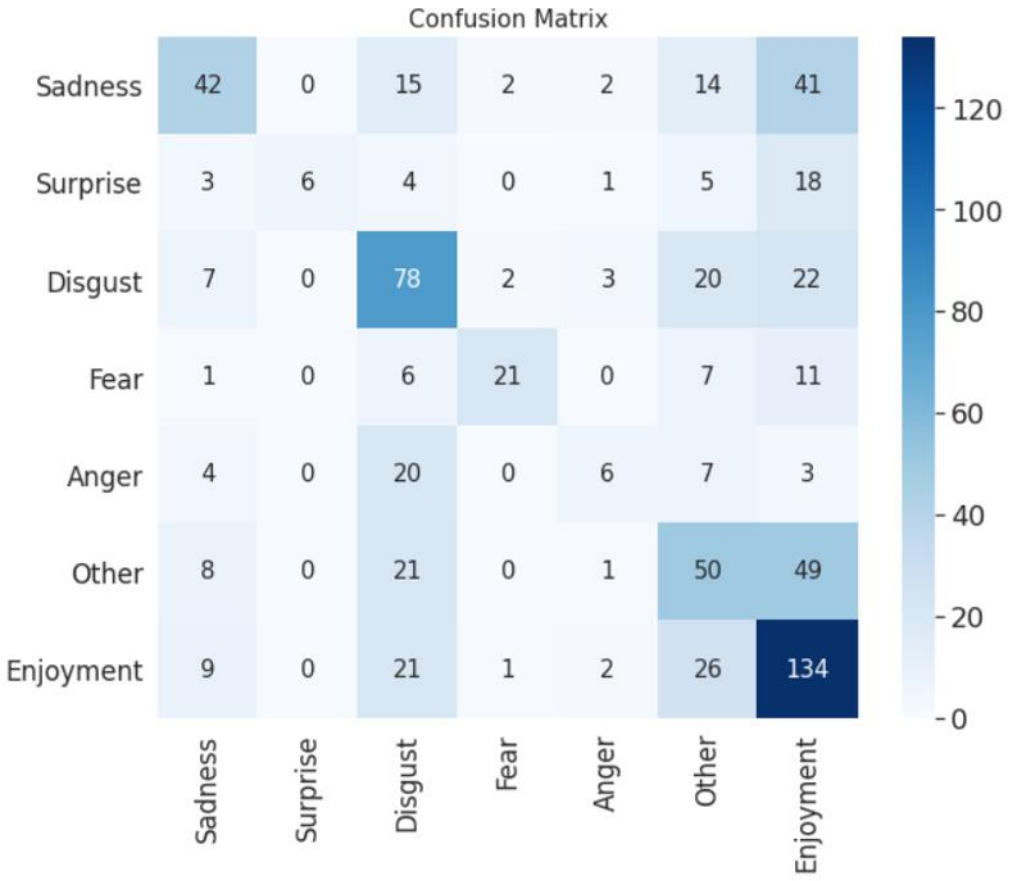
Tập dữ liệu chúng tôi xây dựng là một tập hợp các văn bản và nó là một tập dữ liệu không có cấu trúc, vì vậy rất khó để tiếp cận với các mô hình lần đầu tiên. Vì vậy, chúng tôi thực hiện một số phương pháp tiền xử lý dữ liệu, những phương pháp phổ biến trong NLP. Đầu tiên, chúng tôi mã hóa các nhận xét đó trong tập dữ liệu. Không giống như tiếng Anh hay các ngôn ngữ phổ biến khác, "dấu cách" trong tiếng Việt là dấu hiệu phân cách âm tiết, không phải từ. Đó là lý do tại sao mã hóa văn bản rất quan trọng và nó ảnh hưởng trực tiếp đến kết quả của các mô hình. Ở phương pháp Logistic Regression chúng tôi sử dụng chức năng ViTokenizer của thư viện pyvi [11], chuyên về NLP tiếng Việt để mã hóa văn bản. Sau đó chúng tôi tiến hành xóa các ký tự đặc biệt và các số. Cuối cùng chúng tôi tiến hành xóa các stopword [12].

5.2 Kết quả thực nghiệm

5.2.1 Logistic Regression

a) UIT-VSMEC [1]

– Confusion matrix.



Hình 5.1: Confusion Matrix trên bộ UIT-VSMEC khi chạy bằng Logistic Regression

+ Nhãn Sadness dễ bị nhận nhầm là nhãn Disgust, Other, Enjoyment.
+ Nhãn Surprise dễ bị nhận nhầm là nhãn Enjoyment.
+ Nhãn Disgust dễ bị nhận nhầm là nhãn Other và Enjoyment.
+ Nhãn Fear dễ bị nhận nhầm là nhãn Enjoyment.
+ Nhãn Anger dễ bị nhận nhầm là nhãn Disgust.
+ Nhãn Other dễ bị nhận nhầm là nhãn Disgust và Enjoyment.
+ Nhãn Enjoyment dễ bị nhận nhầm là nhẫn Disgust và Other.

– Kết quả đánh giá trên các độ đo.

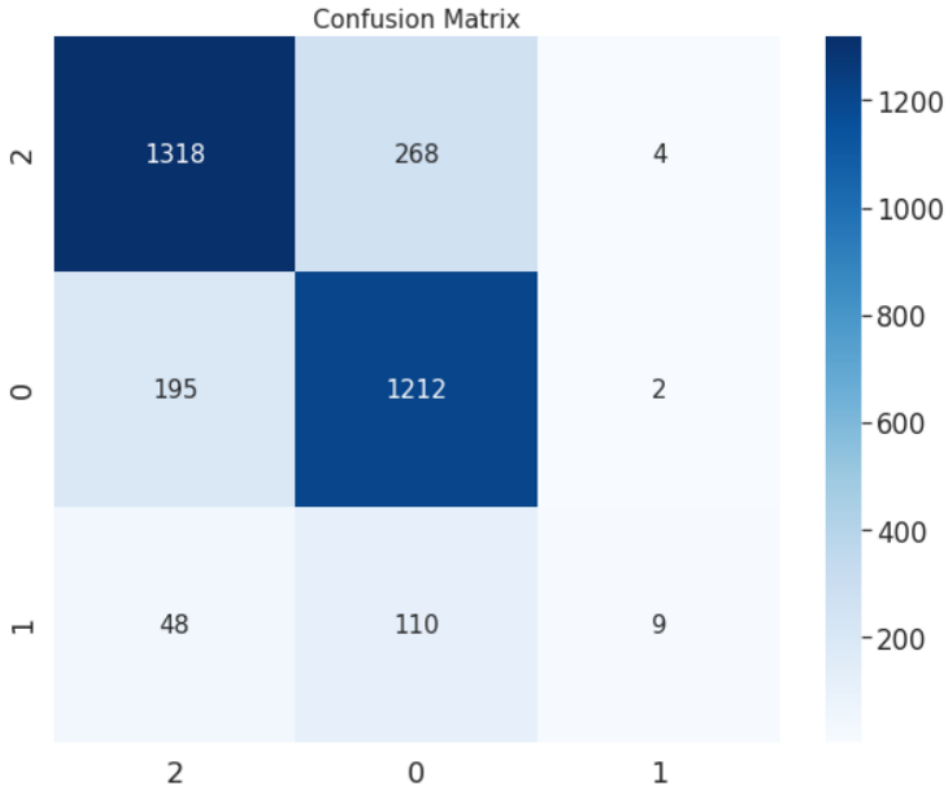
```
accuracy: 0.4862914862914863
F1 - macro: 0.42922024710492696

Classification report in Training set
              precision    recall  f1-score   support

     Sadness       0.57      0.36      0.44       116
    Surprise       1.00      0.16      0.28        37
     Disgust       0.47      0.59      0.53       132
        Fear       0.81      0.46      0.58        46
       Anger       0.40      0.15      0.22        40
       Other       0.39      0.39      0.39       129
   Enjoyment       0.48      0.69      0.57       193

    accuracy                           0.49       693
   macro avg       0.59      0.40      0.43       693
weighted avg       0.52      0.49      0.47       693
```

Hình 5.2: Kết quả đánh giá các độ đo trên bộ UIT-VSMEC khi chạy bằng Logistic Regression

b) UIT-VSFC [2]

– Nhãn **Sentiment**.

+ Confusion matrix.



Hình 5.3: Confusion Matrix trên bộ UIT-VSFC trên nhãn Sentiment khi chạy bằng Logistic Regression

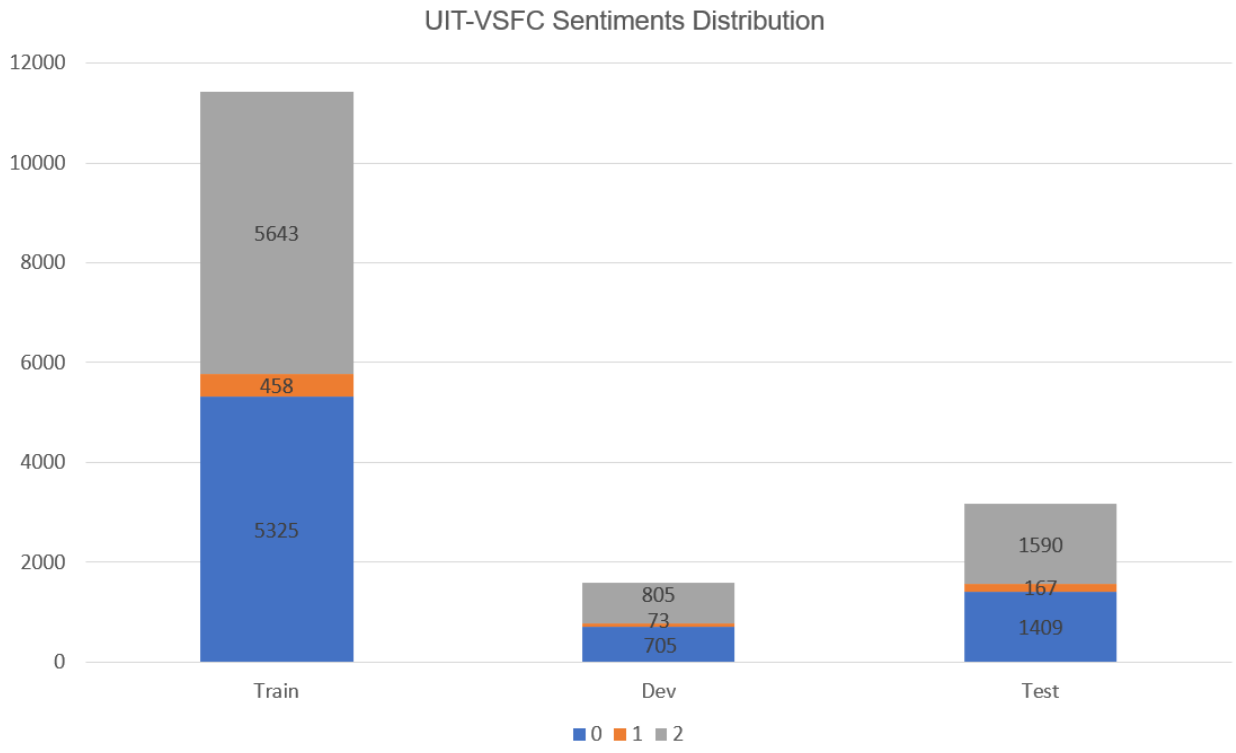
- Ở đây ta thấy nhãn số 2 dễ bị nhận nhầm sang nhãn 0, nhãn số 0 dễ bị nhận nhầm là nhãn số 2, nhãn số 1 dễ bị nhận nhầm là nhãn 0 và 2.

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.8019583070120025
F1 - macro: 0.5812434481069292

Classification report in Training set
              precision    recall  f1-score   support

           2       0.84      0.83      0.84      1590
           0       0.76      0.86      0.81      1409
           1       0.60      0.05      0.10       167

    accuracy                           0.80      3166
   macro avg       0.74      0.58      0.58      3166
weighted avg       0.79      0.80      0.79      3166
```

Hình 5.4: Kết quả đánh giá các độ đo trên bộ UIT-VSFC trên nhãn Sentiment khi chạy bằng Logistic Regression

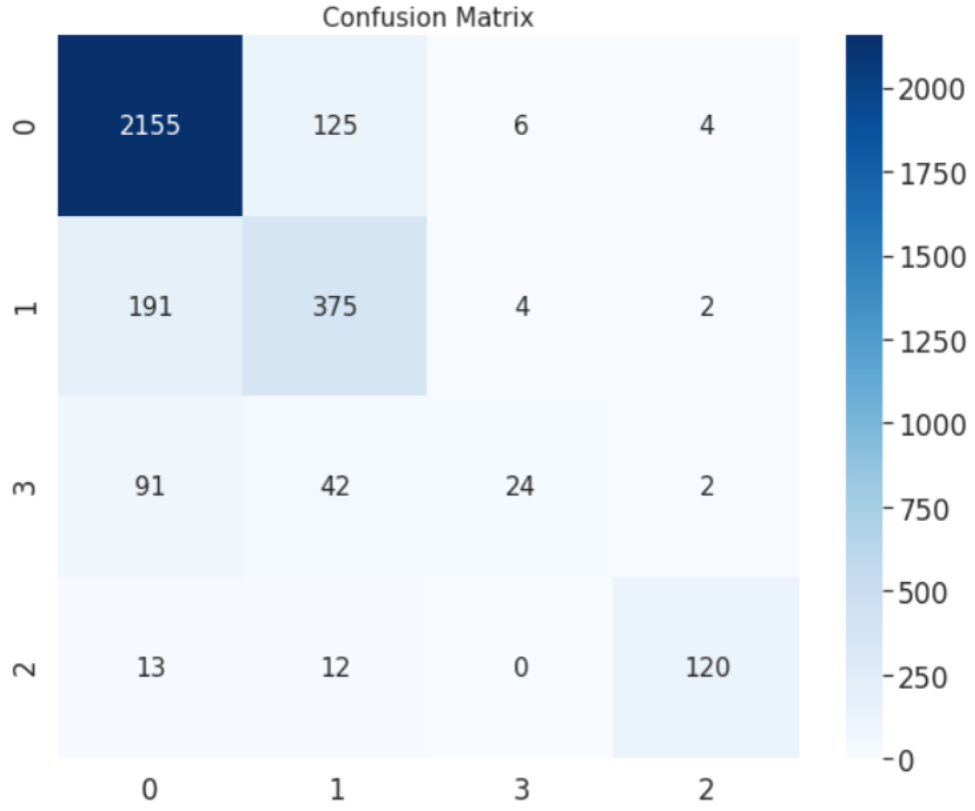
- Độ chính xác của nhãn 1 thấp hơn nhãn 0 và 2.
- Nhìn vào bảng đánh giá, ta thấy rằng độ chính xác của nhãn 1 rất thấp. Nguyên nhân là do phân phối của nhãn số 1 trên tập train quá ít cho nên mô hình không học được nhiều dữ liệu từ nhãn số 1 dẫn đến dự báo kết quả thấp.



Hình 5.5: Phân phối các nhãn của bộ dữ liệu UIT-VSFC trên nhãn Sentiment

– Nhãn **Topic**.

+ Confusion matrix.



Hình 5.6: Confusion Matrix trên bộ UIT-VSFC trên nhãn Topic khi chạy bằng Logistic Regression

- Nhãn số 3 bị nhầm lẫn rất nhiều.

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.8445988629185092
F1 - macro: 0.6757957107776238

Classification report in Training set
              precision    recall  f1-score   support

           0       0.88      0.94      0.91      2290
           1       0.68      0.66      0.67       572
           3       0.71      0.15      0.25       159
           2       0.94      0.83      0.88       145

    accuracy                           0.84      3166
   macro avg       0.80      0.64      0.68      3166
weighted avg       0.84      0.84      0.83      3166
```

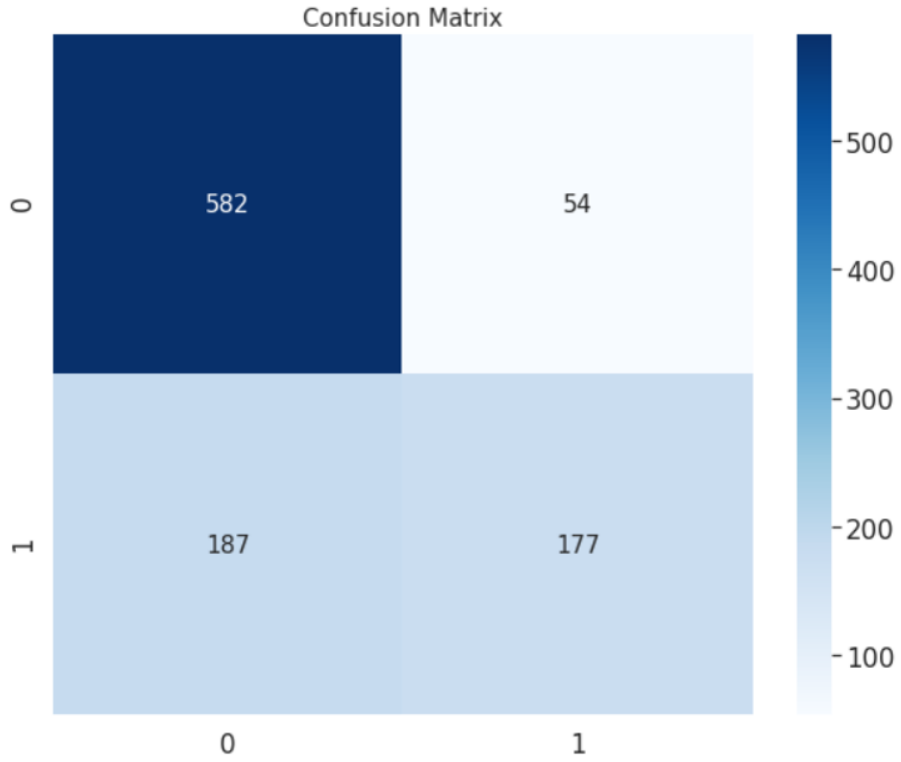
Hình 5.7: Kết quả đánh giá các độ đo trên bộ UIT-VSFC trên nhãn Topic khi chạy bằng Logistic Regression

- Nhãn 0 và 2 có độ chính xác cao nhất.

c) UIT-ViCTSD [3]

– Nhãn **Constructiveness**.

+ Confusion matrix.



Hình 5.8: Confusion Matrix trên bộ UIT-ViCTSD trên nhãn Constructiveness khi chạy bằng Logistic Regression

- Nhãn 1 dễ bị nhầm sang nhãn 0.

+ Kết quả đánh giá trên các độ đo.

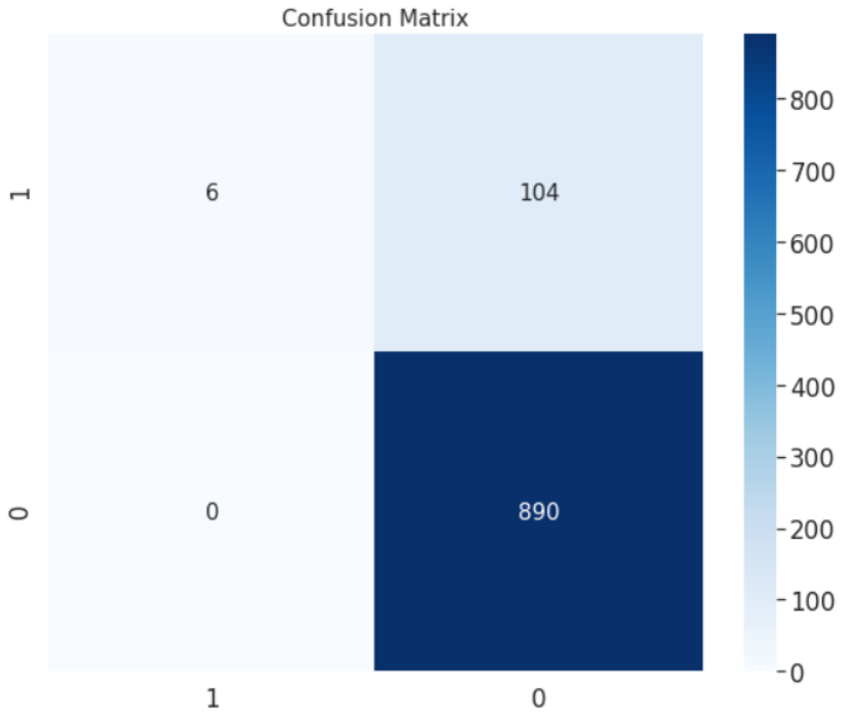
```
accuracy: 0.759
F1 - macro: 0.7117138670414787

Classification report in Training set
              precision    recall  f1-score   support

           0       0.76      0.92      0.83       636
           1       0.77      0.49      0.59       364

    accuracy                           0.76      1000
   macro avg       0.76      0.70      0.71      1000
weighted avg       0.76      0.76      0.74      1000
```

Hình 5.9: Kết quả đánh giá các độ đo trên bộ UIT-ViCTSD trên nhãn Constructiveness khi chạy bằng Logistic Regression

- Độ chính xác của nhãn 1 thấp nhiều hơn so với nhãn 0.

– Nhãn **Toxicity**.

+ Confusion matrix.



Hình 5.10: Confusion Matrix trên bộ UIT-ViCTSD trên nhãn Toxicity khi chạy bằng Logistic Regression

- Nhãn 1 bị nhận nhầm là nhãn số 0 rất nhiều.

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.896
F1 - macro: 0.5241232886741343

Classification report in Training set
              precision    recall  f1-score   support

           1       1.00      0.05      0.10       110
           0       0.90      1.00      0.94       890

    accuracy                           0.90      1000
   macro avg       0.95      0.53      0.52      1000
weighted avg       0.91      0.90      0.85      1000
```

Hình 5.11: Kết quả đánh giá các độ đo trên bộ UIT-ViCTSD trên nhãn Toxicity khi chạy bằng Logistic Regression

- Độ chính xác của nhãn số 1 rất thấp.

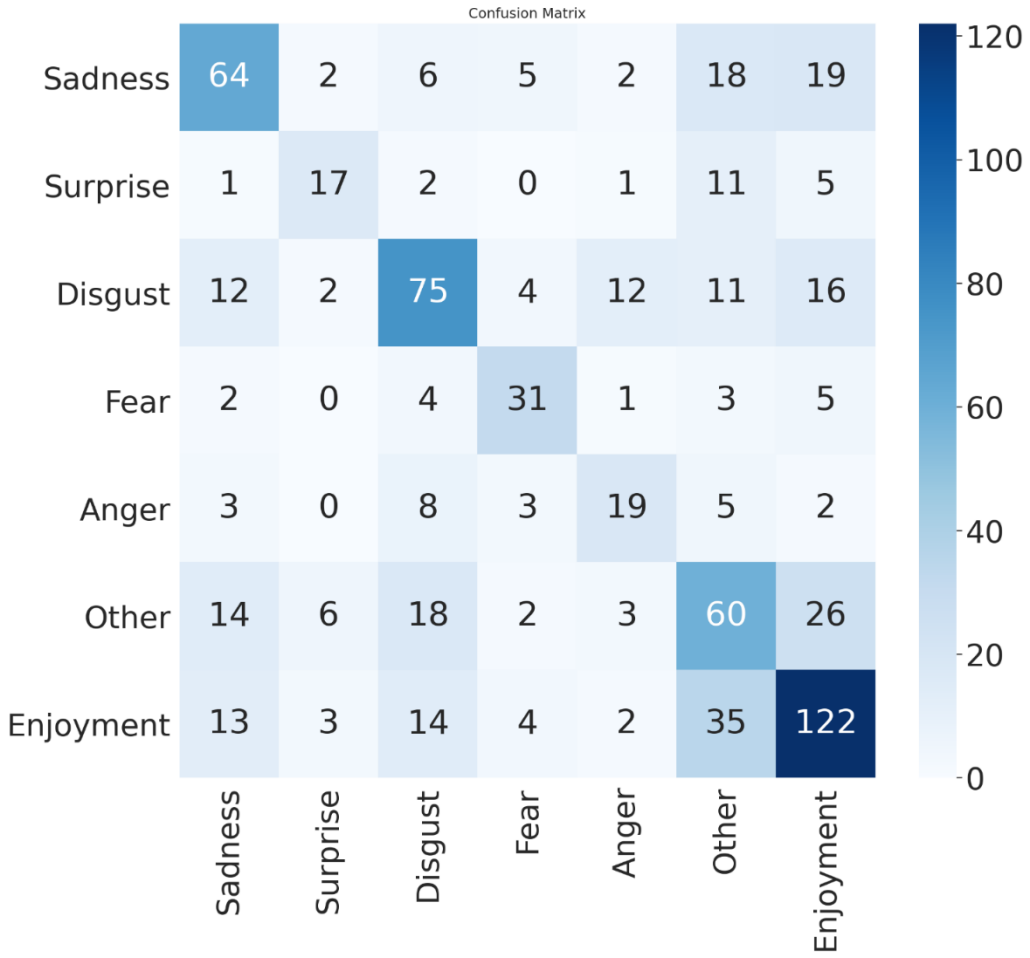
5.2.2 CNN-LSTM

Trong phương pháp này, chúng tôi sử dụng lớp Embedding cho tập từ vựng được lấy ra theo từng bộ ngữ liệu. Quá trình trích xuất tập từ vựng có sử dụng cùng bộ Stop Words tại mục 5.1.

a) UIT-VSMEC [1]

– Confusion matrix.

+ Tiến hành đánh giá kết quả của mô hình trên tập test của bộ ngữ liệu UIT-VSMEC chúng tôi thu được Confusion Matrix như sau:



Hình 5.12: Confusion Matrix trên bộ UIT-VSMEC khi chạy bằng CNN-LSTM

+ Dựa vào kết quả trên, ta thấy lớp Enjoyment thường bị nhận nhầm là lớp Other, để giải thích vấn đề này, chúng tôi tiến hành đánh giá bằng những độ đo khác.

– Kết quả đánh giá trên các độ đo.

+ Chúng tôi cũng tiến hành đánh giá trên các độ đo khác nhau để biết cụ thể mô hình còn hạn chế trong việc phân loại ở những lớp nào.

```
accuracy: 0.5598845598845599
F1 - macro: 0.5504537405033828
Classification report in Training set
              precision    recall  f1-score   support

     Sadness       0.59      0.55      0.57       116
    Surprise       0.57      0.46      0.51        37
     Disgust       0.59      0.57      0.58       132
        Fear       0.63      0.67      0.65        46
       Anger       0.47      0.47      0.48        40
       Other       0.42      0.47      0.44       129
   Enjoyment       0.63      0.63      0.63       193

    accuracy                           0.56       693
   macro avg       0.56      0.55      0.55       693
weighted avg       0.56      0.56      0.56       693
```

Hình 5.13: Kết quả đánh giá các độ đo trên bộ UIT-VSMEC khi chạy bằng CNN-LSTM

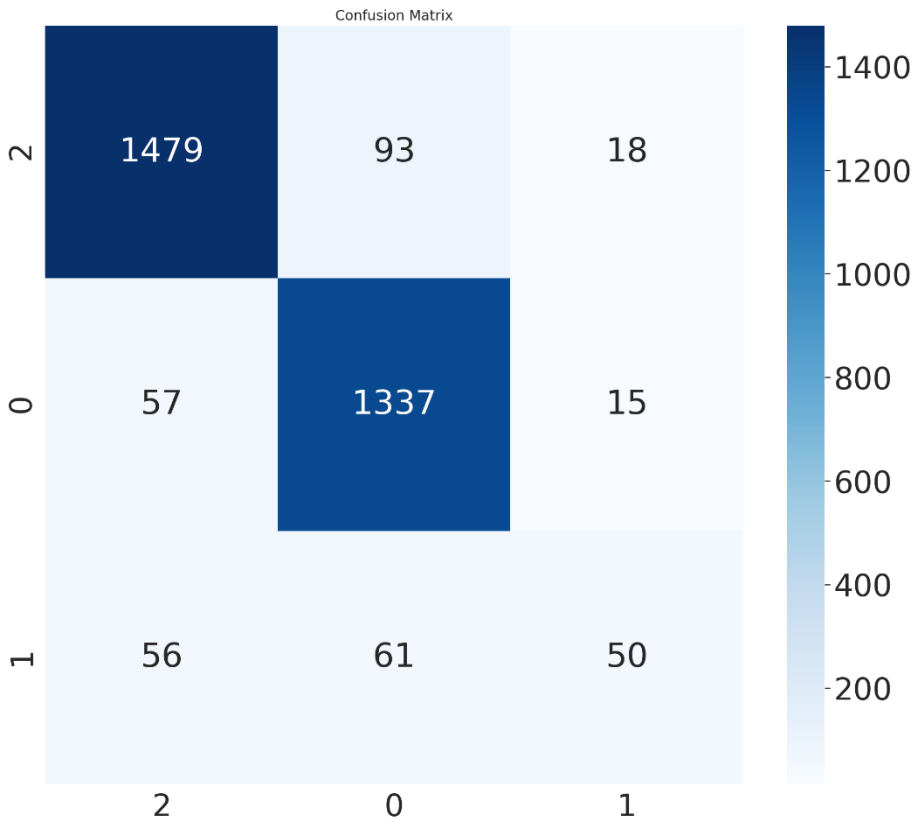
+ Độ chính xác trên các lớp gần như tương đồng nhau, tuy nhiên lỗi vẫn còn khá nhiều ở lớp Other và Enjoyment kéo theo độ chính xác giảm nhiều. Với lớp Other chỉ còn 0.42 trên precision từ đó kéo theo accuracy cũng chỉ ở mức 0.56.

b) UIT-VSFC [2]

– Nhãn **Sentiment**.

+ Confusion matrix.

- Tiến hành đánh giá kết quả của mô hình trên tập test của bộ ngữ liệu UIT-VSFC trên nhãn **Sentiment** chúng tôi thu được Confusion Matrix như sau:



Hình 5.14: Confusion Matrix trên bộ UIT-VSFC trên nhãn Sentiment khi chạy bằng CNN-LSTM

- Các lớp bị nhầm lẫn khá nhiều, đặc biệt là ở lớp 2 thường bị nhận nhầm là lớp 0.

+ Kết quả đánh giá trên các độ đo.

- Chúng tôi cũng tiến hành đánh giá trên các độ đo khác nhau để biết cụ thể mô hình còn hạn chế trong việc phân loại ở những lớp nào.

```
accuracy: 0.9052432090966519
F1 - macro: 0.7505576627148399
Classification report in Training set
              precision    recall  f1-score   support

           2       0.93      0.93      0.93      1590
           0       0.90      0.95      0.92      1409
           1       0.60      0.30      0.40       167

    accuracy                           0.91      3166
   macro avg       0.81      0.73      0.75      3166
weighted avg       0.90      0.91      0.90      3166
```

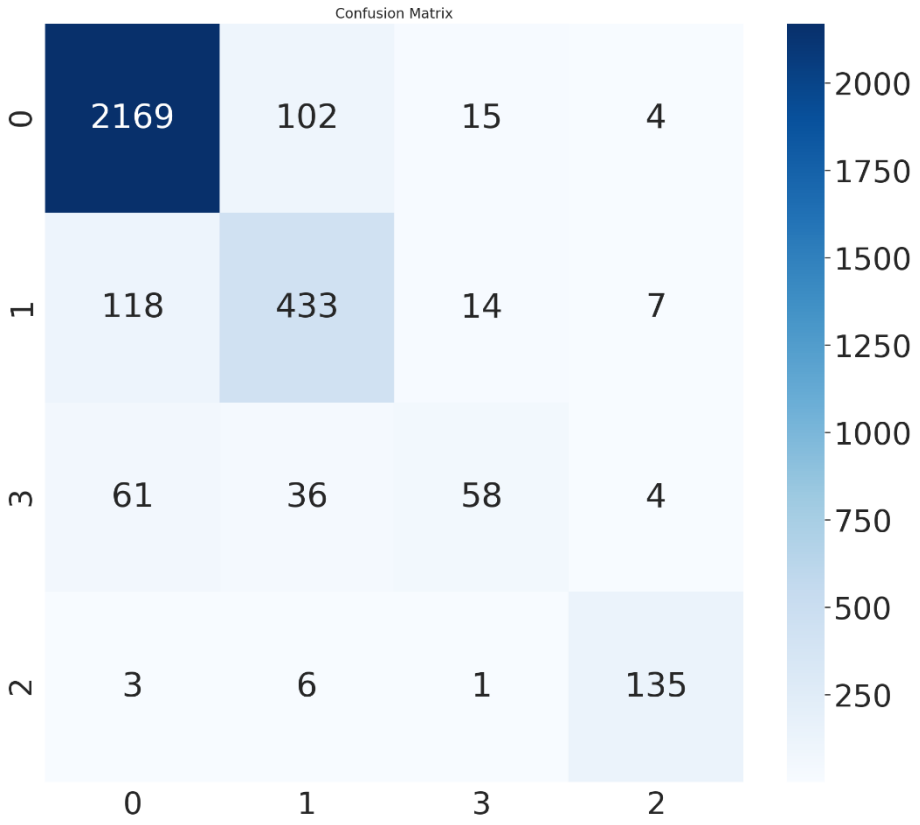
Hình 5.15: Kết quả đánh giá các độ đo trên bộ UIT-VSFC trên nhãn Sentiment khi chạy bằng CNN-LSTM

- Kết quả đánh giá trên các độ đo khác lại cho thầy lớp 1 là lớp có độ chính xác cao hơn, nguyên nhân xuất phát từ số lượng mẫu có nhãn là 1 quá ít, dẫn đến sự nhầm lẫn trên lớp 1 là rất nhiều đối với phân phối của nó, dẫn đến độ chính xác thấp.

– Nhãn **Topic**.

+ Confusion matrix.

- Tiến hành đánh giá kết quả của mô hình trên tập test của bộ ngữ liệu UIT-VSFC trên nhãn Topic chúng tôi thu được Confusion Matrix như sau:



Hình 5.16: Confusion Matrix trên bộ UIT-VSFC trên nhãn Topic khi chạy bằng CNN-LSTM

- Nhãn số 3 thường bị nhận nhầm thành 0 và 1, con số này lại rất lớn so với phân phối của nhãn số 3 mặc dù nhãn số 0 lại có tổng lỗi lớn hơn. Giải thích cho điều này chúng tôi tiến hành đánh giá trên các độ đo khác.

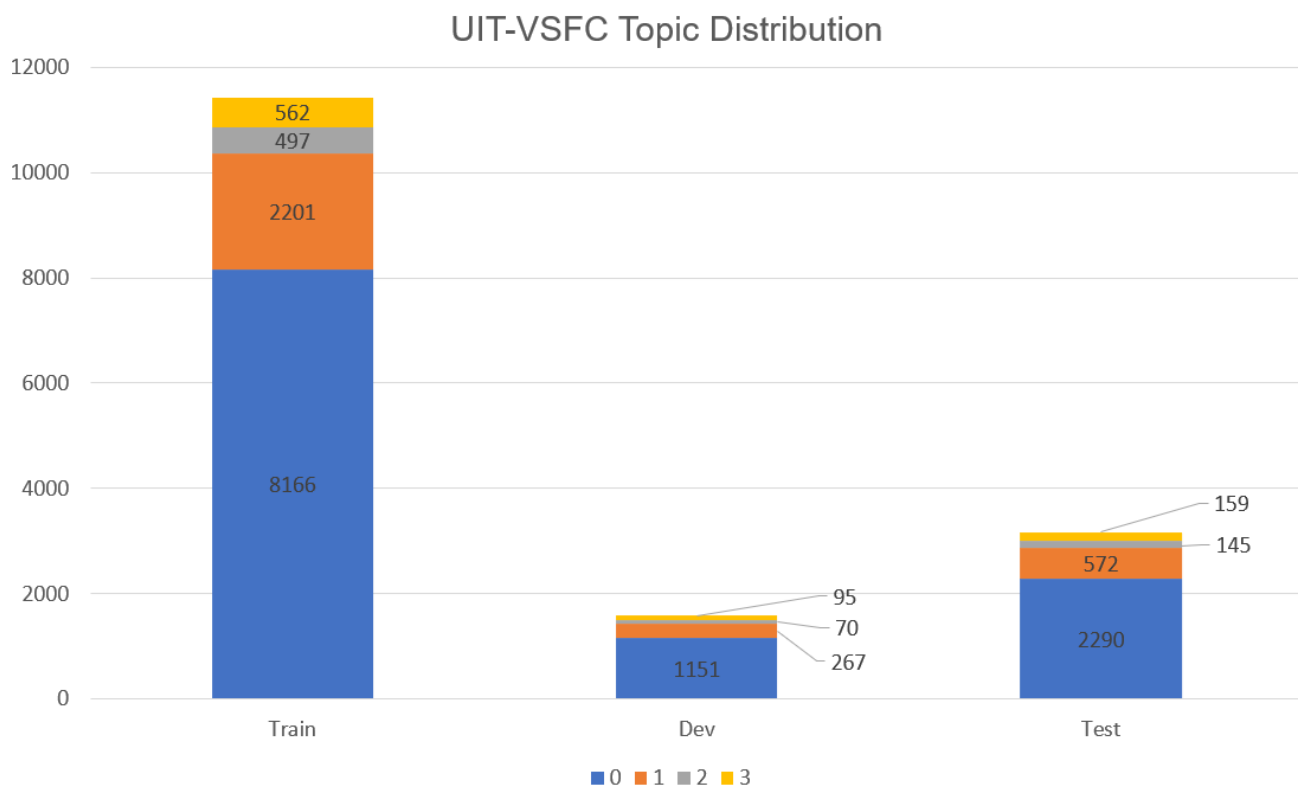
+ Kết quả đánh giá trên các độ đo.

- Chúng tôi cũng tiến hành đánh giá trên các độ đo khác nhau để biết cụ thể mô hình còn hạn chế trong việc phân loại ở những lớp nào.

```
accuracy: 0.8828174352495262
F1 - macro: 0.7683252699692319
Classification report in Training set
              precision    recall  f1-score   support

           0       0.92      0.95      0.93      2290
           1       0.75      0.76      0.75       572
           3       0.66      0.36      0.47       159
           2       0.90      0.93      0.92       145

    accuracy                           0.88      3166
   macro avg       0.81      0.75      0.77      3166
weighted avg       0.88      0.88      0.88      3166
```

Hình 5.17: Kết quả đánh giá các độ đo trên bộ UIT-VSFC trên nhãn Topic khi chạy bằng CNN-LSTM

- Kết quả cho thấy lớp 3 có độ chính xác thấp nhất, nguyên nhân đến từ phân phối của nhãn số 3 ở tập train và số lượng mẫu trên tập test của nó. Từ đó dẫn đến việc mô hình chưa học đủ nhiều cho nhãn số 3 để dự đoán đúng, hơn nữa với số lượng mẫu ít trên tập test dẫn đến việc khi dự báo dù số lượng mẫu dự đoán sai là không đáng kể so với nhãn số 1 nhưng sẽ ảnh hưởng rất lớn đến độ chính xác của nhãn số 3.



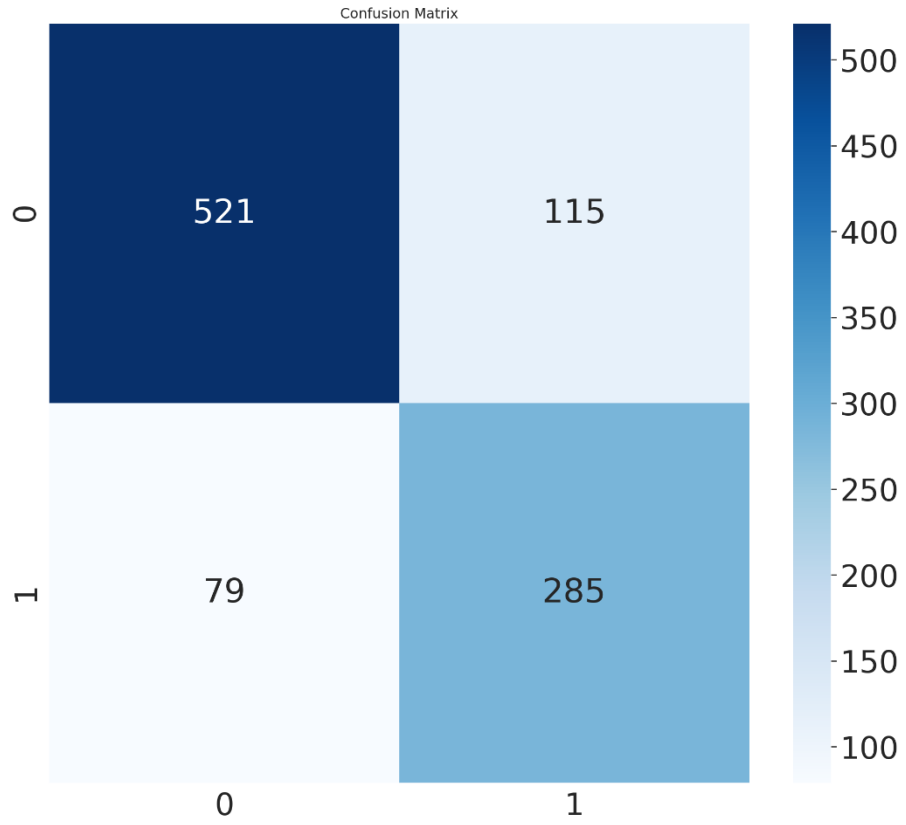
Hình 5.18: Phân phối các nhãn của bộ dữ liệu UIT-VSFC trên nhãn Topic

c) UIT-ViCTSD [3]

– Nhãn **Constructiveness**.

+ Confusion matrix.

- Tiến hành đánh giá kết quả của mô hình trên tập test của bộ ngữ liệu UIT-ViCTSD trên nhãn Constructiveness chúng tôi thu được Confusion Matrix như sau:



Hình 5.19: Confusion Matrix trên bộ UIT-ViCTSD trên nhãn Constructiveness khi chạy bằng CNN-LSTM

- Số lượng mẫu dự đoán sai trên nhãn 0 nhiều hơn nhãn 1, tuy nhiên về phân phối thì số lượng nhãn 0 lại nhiều hơn, để đánh giá chi tiết hơn chúng tôi tiến hành đánh giá trên các độ đo khác.

+ Kết quả đánh giá trên các độ đo.

- Chúng tôi cũng tiến hành đánh giá trên các độ đo khác nhau để biết cụ thể mô hình còn hạn chế trong việc phân loại ở những lớp nào.

```
accuracy: 0.806
F1 - macro: 0.7945576848133651
Classification report in Training set
              precision    recall  f1-score   support

           0       0.87      0.82      0.84       636
           1       0.71      0.78      0.75       364

    accuracy                           0.81      1000
   macro avg       0.79      0.80      0.79      1000
weighted avg       0.81      0.81      0.81      1000
```

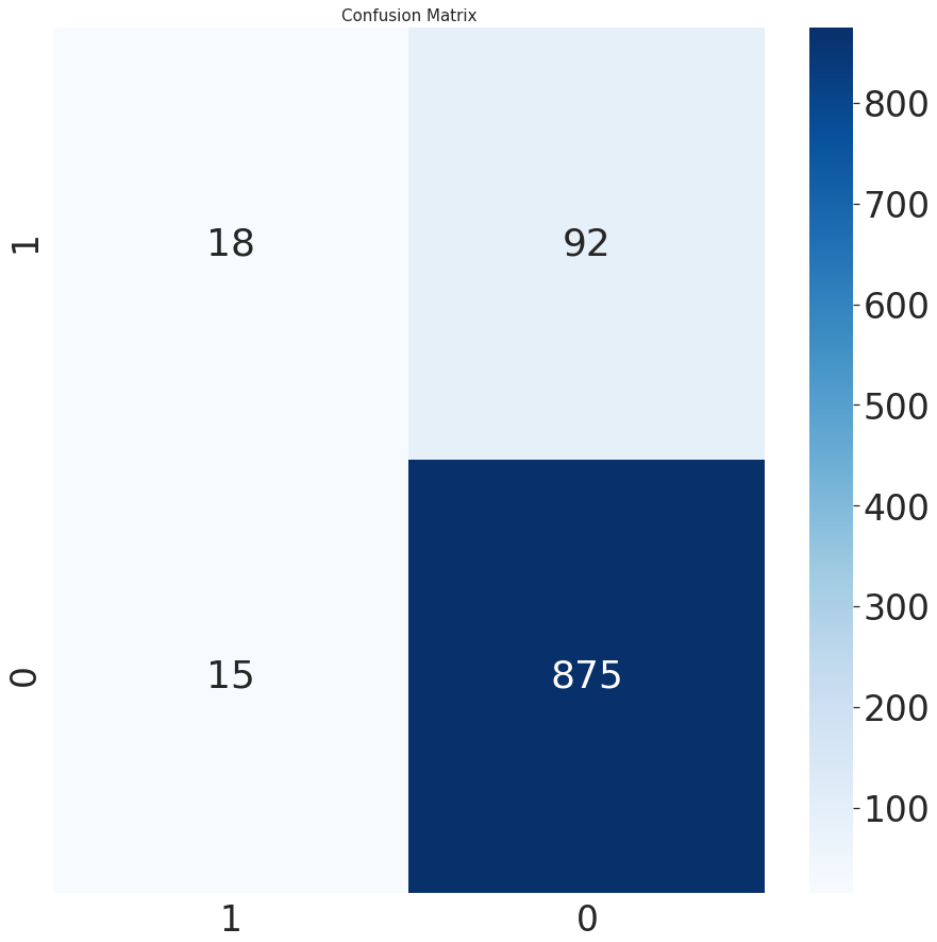
Hình 5.20: Kết quả đánh giá các độ đo trên bộ UIT-ViCTSD trên nhãn Constructiveness khi chạy bằng CNN-LSTM

- Trái ngược với kết quả từ Confusion Matrix, nhãn số 0 lại có độ chính xác cao hơn, nguyên nhân đến từ phân phối của nó ở các tập train và test.

– Nhãn **Toxicity**.

+ Confusion matrix.

- Tiến hành đánh giá kết quả của mô hình trên tập test của bộ ngữ liệu UIT-ViCTSD trên nhãn Toxicity chúng tôi thu được Confusion Matrix như sau:



Hình 5.21: Confusion Matrix trên bộ UIT-ViCTSD trên nhãn Toxicity khi chạy bằng CNN-LSTM

- Nhãn 1 có phân phối ít lại có số mẫu dự đoán sai rất nhiều 92/110.

+ Kết quả đánh giá trên các độ đo.

- Chúng tôi cũng tiến hành đánh giá trên các độ đo khác nhau để biết cụ thể mô hình còn hạn chế trong việc phân loại ở những lớp nào.

```
accuracy: 0.893
F1 - macro: 0.5970642174196293
Classification report in Training set
              precision    recall  f1-score   support

           1       0.55      0.16      0.25       110
           0       0.90      0.98      0.94       890

    accuracy                           0.89      1000
   macro avg       0.73      0.57      0.60      1000
weighted avg       0.87      0.89      0.87      1000
```

Hình 5.22: Kết quả đánh giá các độ đo trên bộ UIT-ViCTSD trên nhãn Toxicity khi chạy bằng CNN-LSTM

- Kết quả đánh giá đúng với Confusion Matrix, độ chính xác trên nhãn 1 rất thấp chỉ 0.55 với precision và 0.16 với recall dẫn đến F1 chỉ còn 0.59.

5.2.3 PhoBert

a) Quy trình huấn luyện dữ liệu

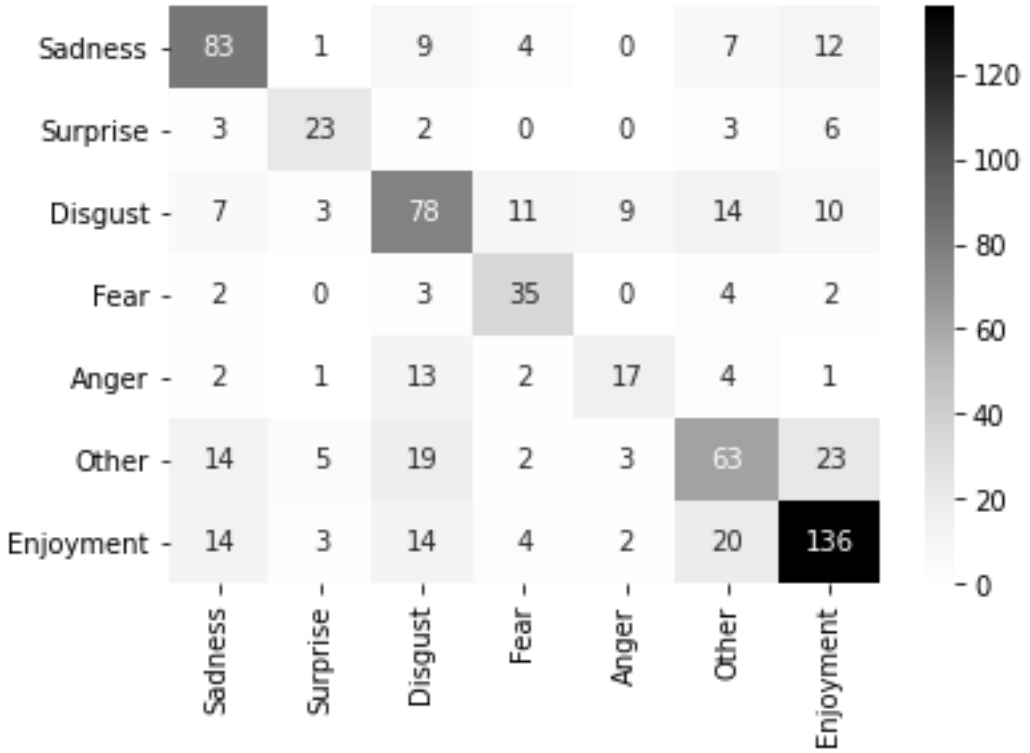
Ở đây, vì tài nguyên của Google Colab là có hạn, vậy nên chúng tôi sẽ train nhiều lần, với lần train sau lấy từ checkpoint của lần train trước có training loss nhỏ nhất, với weight_decay giảm dần sau mỗi lần train. Bảng sau đây thống kê qua các lần train:

Bộ dữ liệu	Label	Lần train	Loss thấp nhất	Checkpoint	Epoch	Batch_size	Weight_Decay	Accuracy	F1 - macro
VSMEC	Emotion	**1**	**0.001**	**16500**	**100**	**32**	**0.01**	**0.6277056**	**0.61074826**
		2	0.0004	17000	100	32	0.001	0.6219336	0.599811668
		3	0.0002	15500	100	32	0.0001	0.6060606	0.585727985
VSFC	Sentiment	**1**	**0.0007**	**8500**	**25**	**32**	**0.01**	**0.9311434**	**0.823251203**
		2	0.0002	13000	40	32	0.001	0.9289324	0.811153827
		3	0.0002	16000	50	32	0.0001	0.9251421	0.807013797
	Topics	**1**	**0.0005**	**15000**	**50**	**32**	**0.01**	**0.8840809**	**0.783222044**
		2	0.0003	14000	50	32	0.001	0.8774479	0.776524034
		3	0.0004	17000	50	32	0.0001	0.8809223	0.782982275
ViCTSD	nstructiven	**1**	**0.0013**	**6000**	**30**	**32**	**0.01**	**0.654**	**0.395405079**
		2	0.0011	6500	30	32	0.001	0.621	0.383096854
		3	0.0007	9500	50	32	0.0001	0.639	0.389871873
	Toxicity	1	0.0005	7000	40	32	0.01	0.895	0.693232714
		2	**0.0005**	**6500**	**50**	**32**	**0.001**	**0.906**	**0.729212758**
		3	0.0208	10000	50	32	0.0001	0.885	0.676230535

Hình 5.23: Quy trình huấn luyện dữ liệu bằng PhoBert

b) UIT-VSMEC [1]

– Confusion matrix.



Hình 5.24: Confusion Matrix trên bộ UIT-VSMEC khi chạy bằng PhoBert

Nhìn vào confusion matrix ta thấy:

+ Nhãn Other và Enjoyment dễ bị nhầm sang nhãn Disgust và ngược lại.

+ Ngoài ra 2 nhãn Other và Enjoyment cũng dễ bị nhầm sang nhãn Sadness.

+ 2 nhãn có độ chính xác cao là Enjoyment và Saddness.

– Kết quả đánh giá trên các độ đo.

```
accuracy: 0.6277056277056277
F1 - macro: 0.6107482596196553
Classification report in Training set
              precision    recall  f1-score   support

     Sadness       0.66      0.72      0.69       116
    Surprise       0.64      0.62      0.63        37
     Disgust       0.57      0.59      0.58       132
        Fear       0.60      0.76      0.67        46
       Anger       0.55      0.42      0.48        40
       Other       0.55      0.49      0.52       129
  Enjoyment        0.72      0.70      0.71       193

    accuracy                           0.63       693
   macro avg       0.61      0.62      0.61       693
weighted avg       0.63      0.63      0.63       693
```

Hình 5.25: Kết quả đánh giá các độ đo trên bộ UIT-VSMEC khi chạy bằng PhoBert

Từ kết quả độ đo ta thấy:

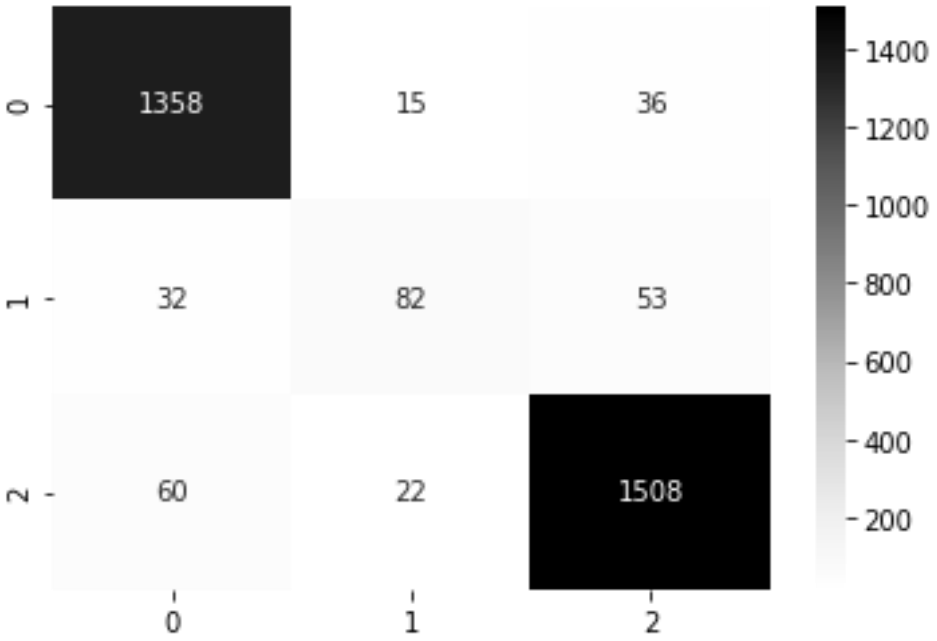
+ Nhìn chung kết quả đánh giá trên các độ đo khá thấp và khá đồng đều trên các nhãn.

+ Đặc biệt, Enjoyment có độ chính xác Precision rất cao (Enjoyment 0.72)

c) UIT-VSFC [2]

– Nhãn **Sentiment**.

+ Confusion matrix.



Hình 5.26: Confusion Matrix trên bộ UIT-VSFC trên nhãn Sentiment khi chạy bằng PhoBert

Dựa vào confusion matrix, ta thấy:

- Ta thấy nhãn 0 và 2 có số lượng nhãn dự đoán đúng rất cao (Nhãn 0: 1358/1409 chiếm 96,3% và nhãn 2: 1507/1590 chiếm 94,7%).
- Trong khi đó, nhãn 1 thì khá thấp (82/167 chiếm 49,1%).

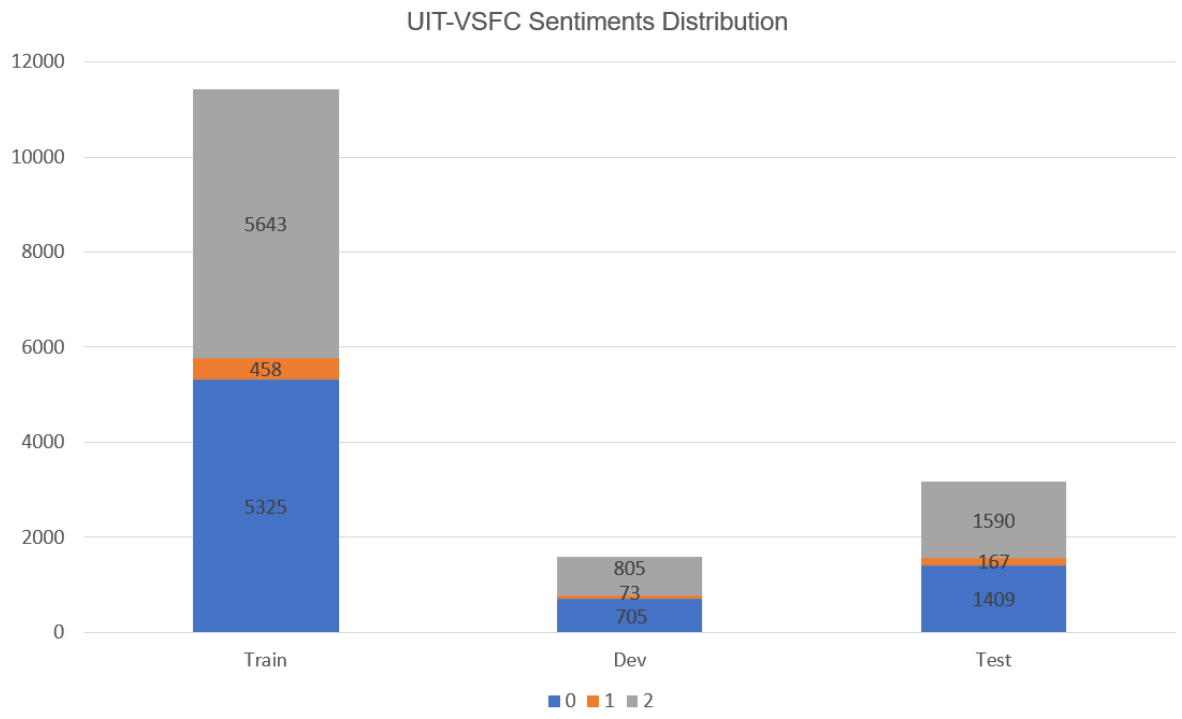
+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.9317751105495894
F1 - macro: 0.8231377193278845
Classification report in Training set
              precision    recall  f1-score   support

           2       0.94      0.95      0.95      1590
           0       0.94      0.96      0.95      1409
           1       0.68      0.49      0.57       167

    accuracy                           0.93      3166
   macro avg       0.86      0.80      0.82      3166
weighted avg       0.93      0.93      0.93      3166
```

Hình 5.27: Kết quả đánh giá các độ đo trên bộ UIT-VSFC trên nhãn Sentiment khi chạy bằng PhoBert

Dựa vào bảng thống kê độ đo ta thấy:

- Nhìn chung kết quả độ đo khác cao.
- Hai nhãn 2 và 0 có độ chính xác khác cao (0.94 ở cả 2 nhãn), cao hơn hẳn so với nhãn 1 (chỉ có 0.68). Lý giải cho điều này, ta có thể xem phân phối các nhãn:

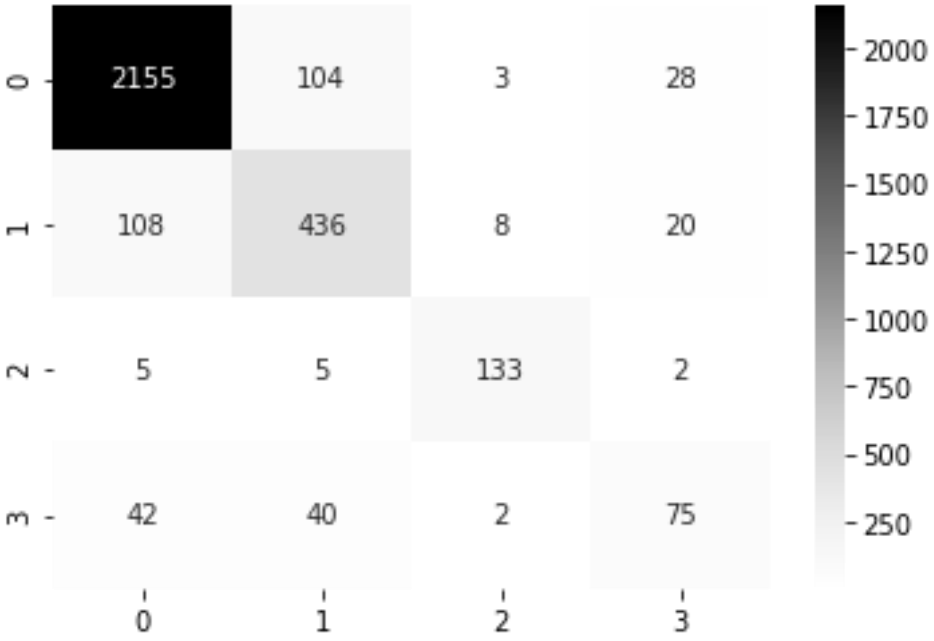


Hình 5.28: Phân phối các nhãn của bộ dữ liệu UIT-VSFC trên nhãn Sentiment

- Theo hình trên, ta thấy nhãn 1 trong cả 3 bộ dữ liệu train, dev và test chiếm tỉ lệ rất thấp so với 2 nhãn còn lại là 0 và 2. Phân phối của nhãn 1 lần lượt là 458/11426 chiếm 4,0% trên tập train và 73/1583 chiếm 4,6% trên tập dev. Điều này dẫn tới việc model không thể học được nhiều như 2 nhãn 0 và 2, từ đó sẽ kéo kết quả dự đoán ở nhãn 1 xuống.

– Nhãn **Topic**.

+ Confusion matrix.



Hình 5.29: Confusion Matrix trên bộ UIT-VSFC trên nhãn Topic khi chạy bằng PhoBert

Qua confusion matrix ta thấy:

- Nhãn 0 có số lượng nhãn dự đoán chính xác rất cao (2155/2290 chiếm 94.1%). Tuy nhiên có đôi lúc nhãn 0 dễ nhận nhầm thành nhãn 1 (104/2290 chiếm 4,5%).
- Nhãn 3 dễ bị nhận nhầm sang nhãn khác (42 nhãn nhận nhầm thành nhãn 0 và 40 nhãn nhận nhầm thành nhãn 1), trong khi đó số nhãn 3 được dự đoán chính xác đạt 75 nhãn.

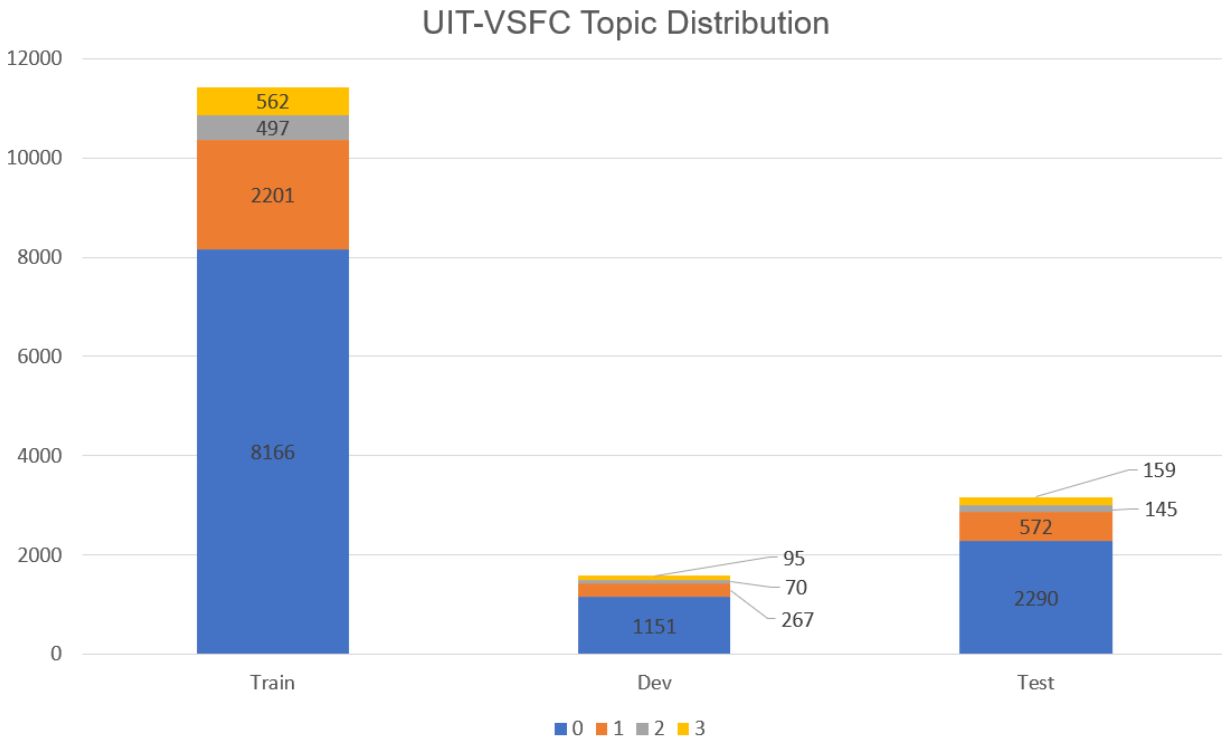
+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.8834491471888819
F1 - macro: 0.7814562463647363
Classification report in Training set
              precision    recall  f1-score   support

           0       0.93      0.94      0.94      2290
           1       0.75      0.76      0.75       572
           3       0.61      0.45      0.52       159
           2       0.91      0.92      0.92       145

    accuracy                           0.88      3166
   macro avg       0.80      0.77      0.78      3166
weighted avg       0.88      0.88      0.88      3166
```

Hình 5.30: Kết quả đánh giá các độ đo trên bộ UIT-VSFC trên nhãn Topic khi chạy bằng PhoBert

Từ bảng thống kê kết quả độ đo ta thấy:

- Hầu hết các nhãn đều có độ chính xác khá cao.
- 2 nhãn 0 và 2 có độ chính xác cao hơn hẳn 2 nhãn còn lại: lần lượt là 0.93 ở nhãn 0 và 0.91 ở nhãn 2 so với 0.75 ở nhãn 1 và 0.61 ở nhãn 3. Để lý giải cho điều này, ta có thể xem phân phối của bộ dữ liệu:



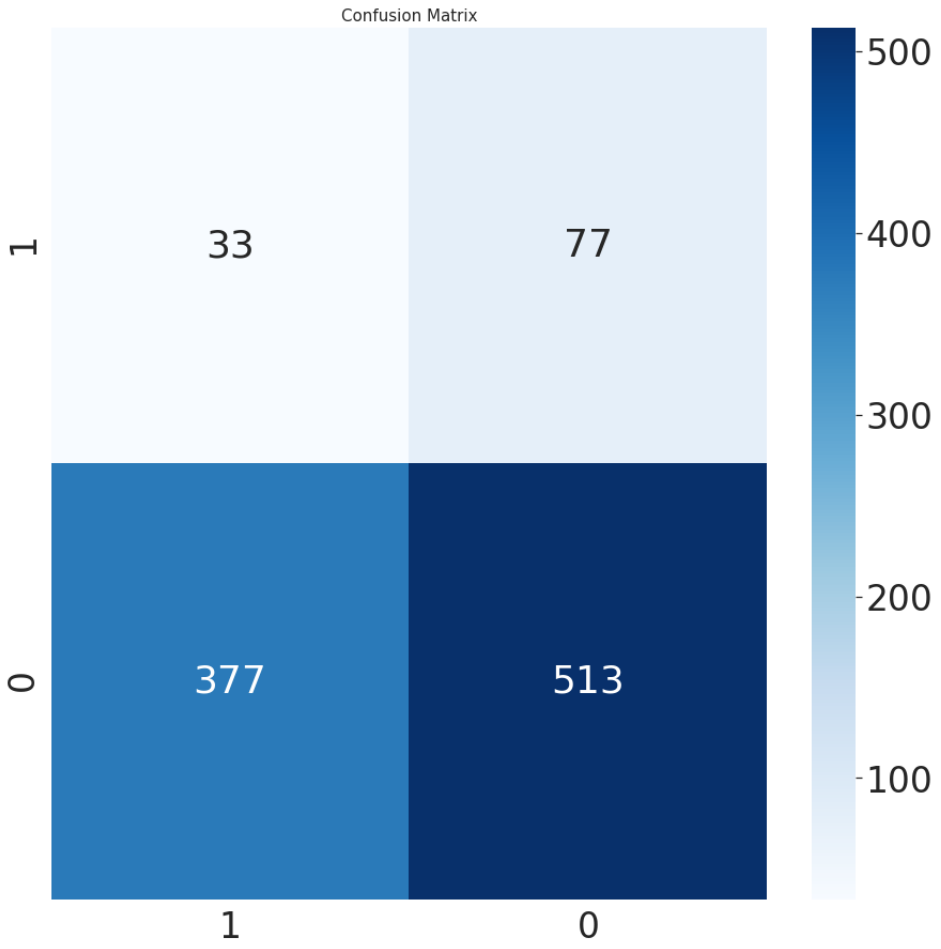
Hình 5.31: Phân phối các nhãn của bộ dữ liệu UIT-VSFC trên nhãn Topic

- Qua hình ảnh trên ta thấy được rằng, nhãn 0 chiếm tỉ lệ rất cao ở trong 3 bộ dữ liệu: lần lượt là 8166/11426 chiếm 71.46% trong tập train, 1151/1538 chiếm 72.7% và 2290/3166 chiếm 72.33%. Từ đây mô hình sẽ học được nhiều dữ liệu từ nhãn 0, vì vậy kết quả độ chính xác trên nhãn này rất cao.

d) UIT-ViCTSD [3]

– Nhãn **Constructiveness**.

+ Confusion matrix.



Hình 5.32: Confusion Matrix trên bộ UIT-ViCTSD trên nhãn Constructiveness khi chạy bằng PhoBert

Qua confusion matrix ta thấy:

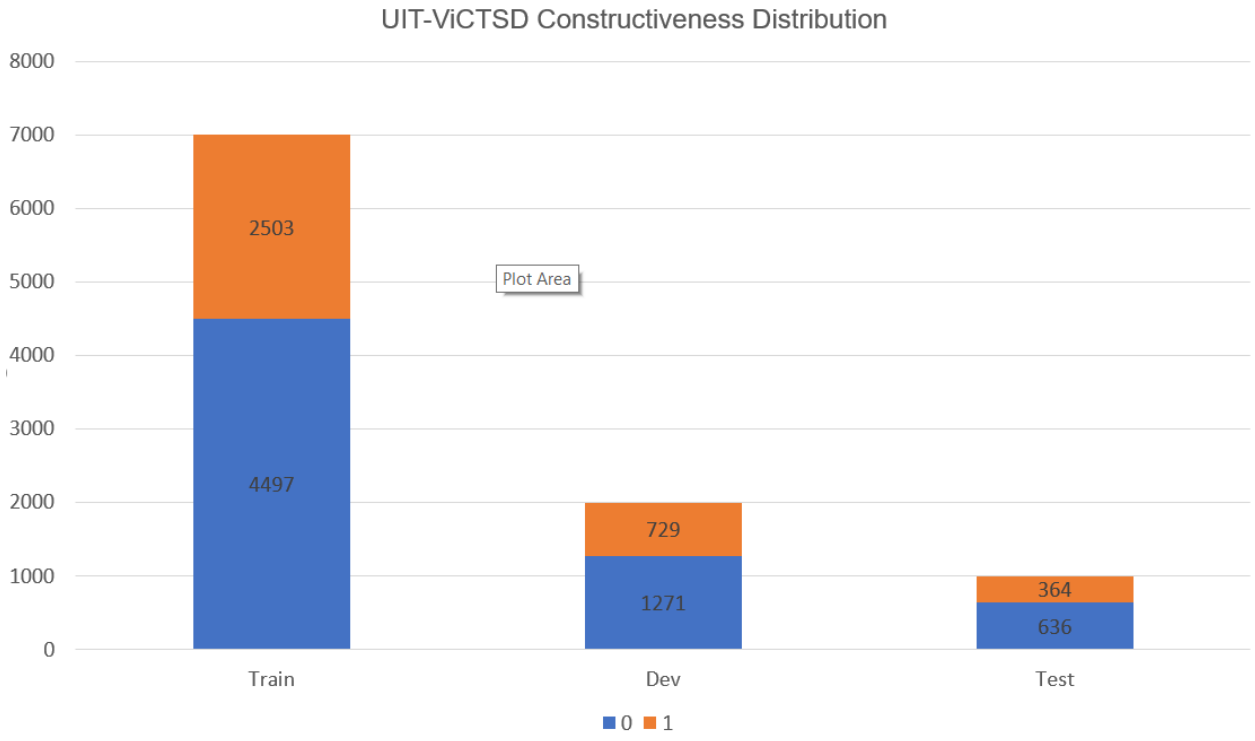
- Nhãn 1 rất dễ nhận nhầm thành nhãn 0 (77/110 nhãn bị nhận nhầm).

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.546
F1 - macro: 0.4100831600831601
Classification report in Training set
              precision    recall  f1-score   support

           1       0.08      0.30      0.13       110
           0       0.87      0.58      0.69       890

    accuracy                           0.55      1000
   macro avg       0.47      0.44      0.41      1000
weighted avg       0.78      0.55      0.63      1000
```

Hình 5.33: Kết quả đánh giá các độ đo trên bộ UIT-ViCTSD trên nhãn Constructiveness khi chạy bằng PhoBert

- Qua kết quả đánh giá các độ đo ta thấy, độ chính xác của nhãn 1 rất thấp: 0.08. Về lý do tại sao nhãn 1 có kết quả thấp như vậy, ta có thể xem qua phân phối các nhãn ở hình dưới:

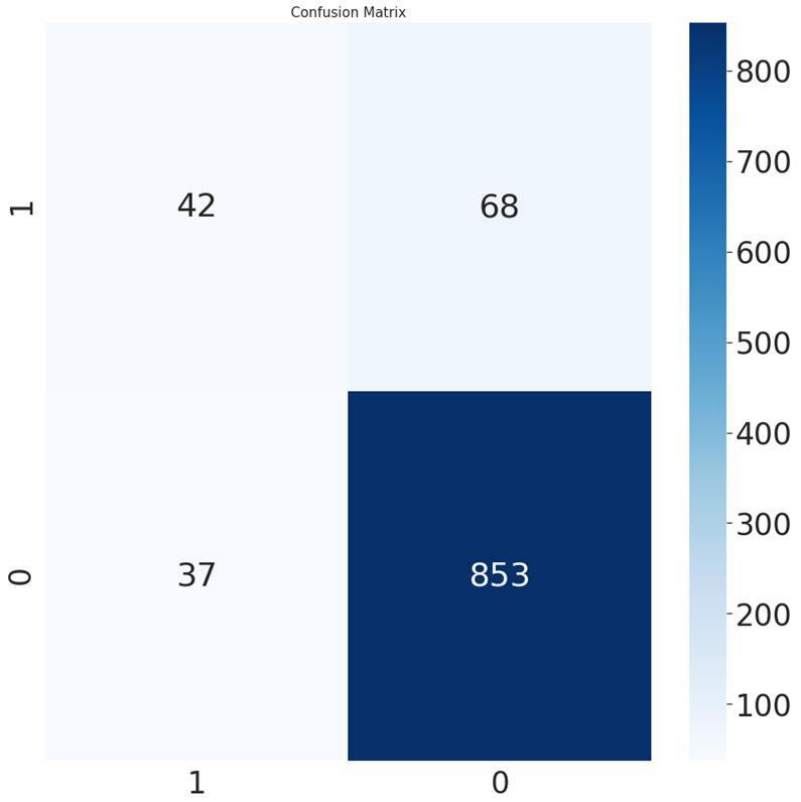


Hình 5.34: Phân phối các nhãn của bộ dữ liệu UIT-ViCTSD trên nhãn Constructiveness

- Qua phân phối chúng ta có thể thấy nhãn 1 có phân phối ít hơn so với nhãn 0 ở trên 3 bộ train, dev, test dẫn đến model của chúng ta sẽ không học được nhiều từ dữ liệu nhãn 1. Ngoài ra, dataset này chủ yếu là bộ dữ liệu thuộc dạng bình luận mạng xã hội, vậy nên dữ liệu sẽ phi cấu trúc, từ đó gây khó khăn cho việc học dữ liệu.

– Nhãn **Toxicity**.

+ Confusion matrix.



Hình 5.35: Confusion Matrix trên bộ UIT-ViCTSD trên nhãn Toxicity khi chạy bằng PhoBert

Qua confusion matrix ta thấy:

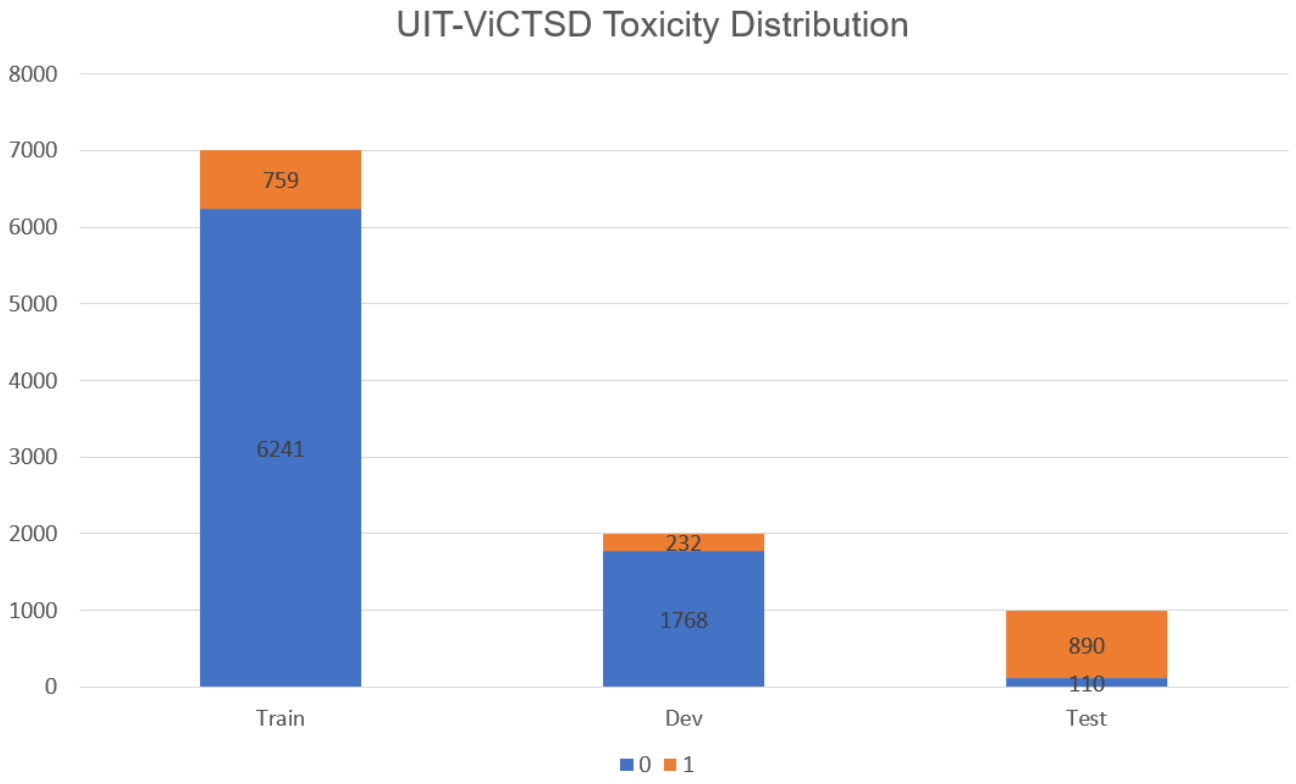
- Nhãn 0 có số lượng nhãn được dự đoán chính xác rất cao. Trong khi đó nhãn 1 thì lại khá thấp mà bị nhận nhầm sang nhãn 0 (68/110 nhãn bị nhận nhầm).

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.906
F1 - macro: 0.7292127581120944
Classification report in Training set
              precision    recall  f1-score   support

           1       0.60      0.45      0.51       110
           0       0.93      0.96      0.95       890

    accuracy                           0.91      1000
   macro avg       0.77      0.70      0.73      1000
weighted avg       0.90      0.91      0.90      1000
```

Hình 5.36: Kết quả đánh giá các độ đo trên bộ UIT-ViCTSD trên nhãn Toxicity khi chạy bằng PhoBert

- Qua kết quả các độ đo ta thấy: Nhãn 0 đạt kết quả cao hơn nhãn 1 vì phân phối dữ liệu của 2 nhãn trên tập dữ liệu không đồng đều (Nhãn 0 nhiều hơn nhãn 1). Ta có thể thấy chi tiết ở hình bên dưới:



Hình 5.37: Phân phối các nhãn của bộ dữ liệu UIT-ViCTSD trên nhãn Toxicity

5.2.4 Bert4News

a) Quy trình huấn luyện dữ liệu

Bert4News [5] [8] cũng là phương pháp State-of-the-art (SOTA) giống PhoBert nên quá trình huấn luận dữ liệu cũng tương tự.

Nhóm cũng tiến hành train 3 lần trên mỗi bộ dữ liệu. Lần train sau sẽ chỉnh sửa Weight_decay giảm xuống. Và lấy epoch có loss thấp nhất để train cho lần tiếp theo. Quá trình train sẽ dừng lại khi kết quả không cải thiện hoặc bị overfit. Khi đó sẽ lấy kết quả của lần train đạt kết quả khả quan nhất.

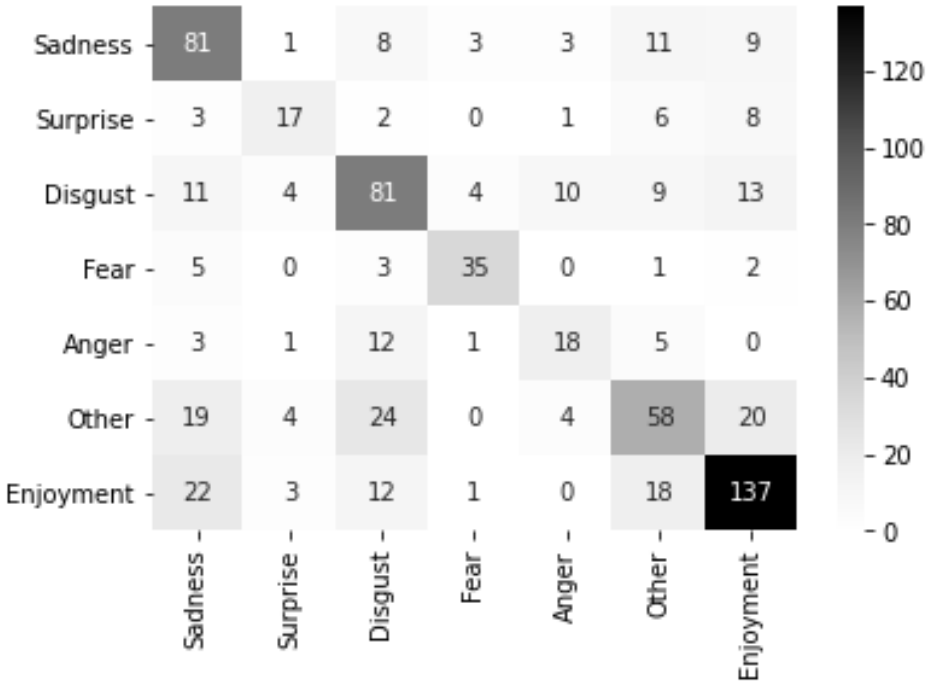
Dưới đây là bảng thống kê kết quả huấn luyện dữ liệu của phương pháp Bert4Bews.

Bộ dữ liệu	Label	Lần train	Loss thấp nhất	Checkpoint	Epoch	Batch_size	Weight_Decay	Accuracy	F1 - macro
VSMEC	Emotion	**1**	**0.0003**	**17000**	**100**	**32**	**0.01**	**0.6161**	**0.5971**
		2	0.0003	17000	100	32	0.001	0.5988	0.5829
		3	0.0004	11500	70	32	0.0001	0.5829	0.5614
VSFC	Sentiment	1	0.0001	35500	100	32	0.01	0.9204	0.7815
		2	**0.0008**	**35500**	**100**	**32**	**0.001**	**0.9264**	**0.7999**
		3	0.0129	24000	70	32	0.0001	0.9223	0.7872
	Topics	**1**	**0.0001**	**29500**	**100**	**32**	**0.01**	**0.88155**	**0.784**
		2	0.0001	34500	100	32	0.001	0.88092	0.781
		3	0.0002	22000	70	32	0.0001	0.88061	0.776
ViCTSD	Constructiveness	1	0.0005	20000	100	32	0.01	0.584	0.4298
		2	0.0039	500	100	32	0.001	0.6	0.4437
		3	**0.0021**	**15000**	**70**	**32**	**0.0001**	**0.619**	**0.4365**
	Toxicity	1	0.0005	12500	100	32	0.01	0.899	0.702
		2	**0.0001**	**500**	**100**	**32**	**0.001**	**0.91**	**0.723**
		3	0.0029	14500	70	32	0.0001	0.895	0.693

Hình 5.38: Quy trình huấn luyện dữ liệu bằng Bert4News

b) UIT-VSMEC [1]

– Confusion matrix.



Hình 5.39: Confusion Matrix trên bộ UIT-VSMEC khi chạy bằng Bert4News

Nhìn vào Confusion matrix trên ta thấy:

+ Nhãn Other dễ nhận nhầm thành nhãn Enjoyment và ngược lại.
+ Nhãn Sadness dễ nhận nhầm thành nhãn Enjoyment, Other và ngược lại.
+ Đặt biệt ta thấy:
  - Nhãn Fear có số lượng nhãn được dự đoán đúng khá cao (35/46 đạt 76,1%).
  - Nhãn Enjoyment có số lượng nhãn được dự đoán đúng cũng khá cao (137/193 đạt 71,0%).

– Kết quả đánh giá trên các độ đo.

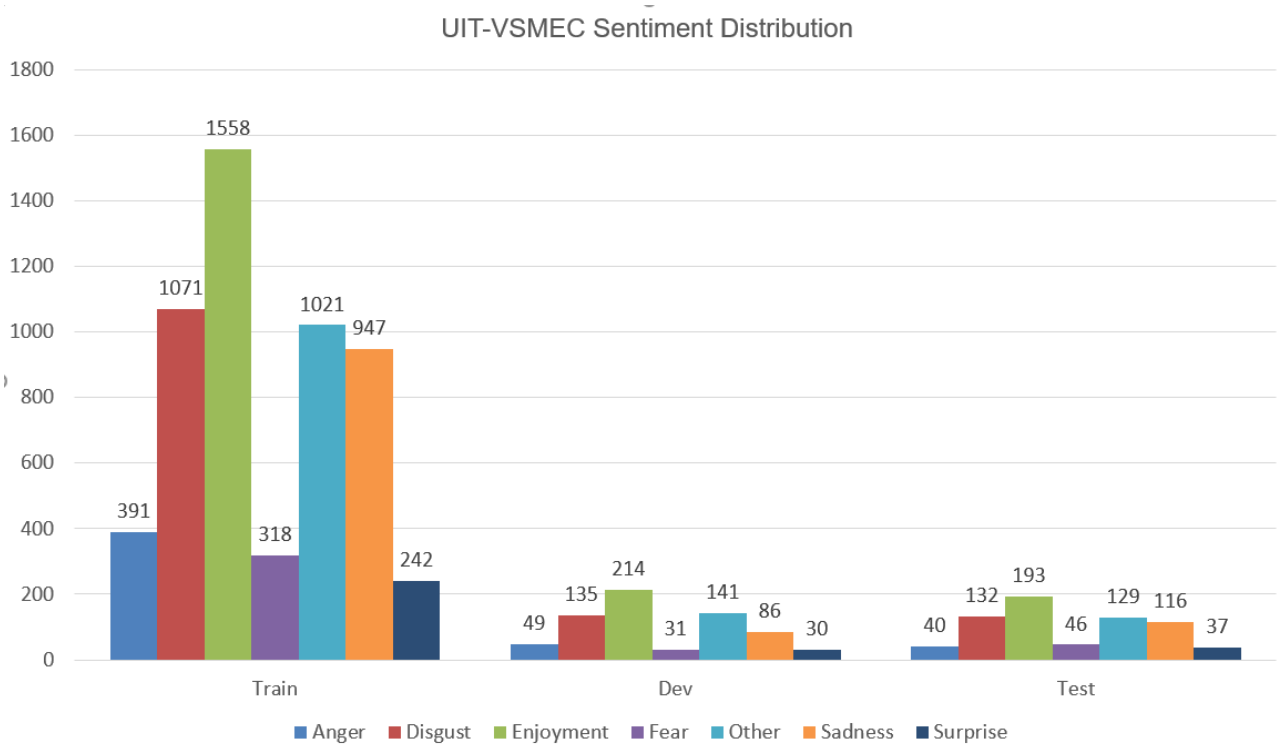
```
accuracy: 0.6161616161616161
F1 - macro: 0.5971387767949762
Classification report in Training set
              precision    recall  f1-score   support

     Sadness       0.56      0.70      0.62       116
    Surprise       0.57      0.46      0.51        37
     Disgust       0.57      0.61      0.59       132
        Fear       0.80      0.76      0.78        46
       Anger       0.50      0.45      0.47        40
       Other       0.54      0.45      0.49       129
   Enjoyment       0.72      0.71      0.72       193

    accuracy                           0.62       693
   macro avg       0.61      0.59      0.60       693
```

Hình 5.40: Kết quả đánh giá các độ đo trên bộ UIT-VSMEC khi chạy bằng Bert4News

Nhìn vào kết quả đánh giá ta thấy:

+ Nhìn chung kết quả đánh giá trên các độ đo khá thấp và khá đồng đều trên các nhãn.
+ Đặc biệt, Nhãn Fear và Enjoyment có độ chính xác Precision rất cao (Fear 0.80 và Enjoyment 0.72)

Về lý do 2 nhãn này đạt được kết quả cao như vậy, ta có thể xem qua phân phối các nhãn ở hình dưới:



Hình 5.41: Phân phối các nhãn của bộ dữ liệu UIT-VSMEC

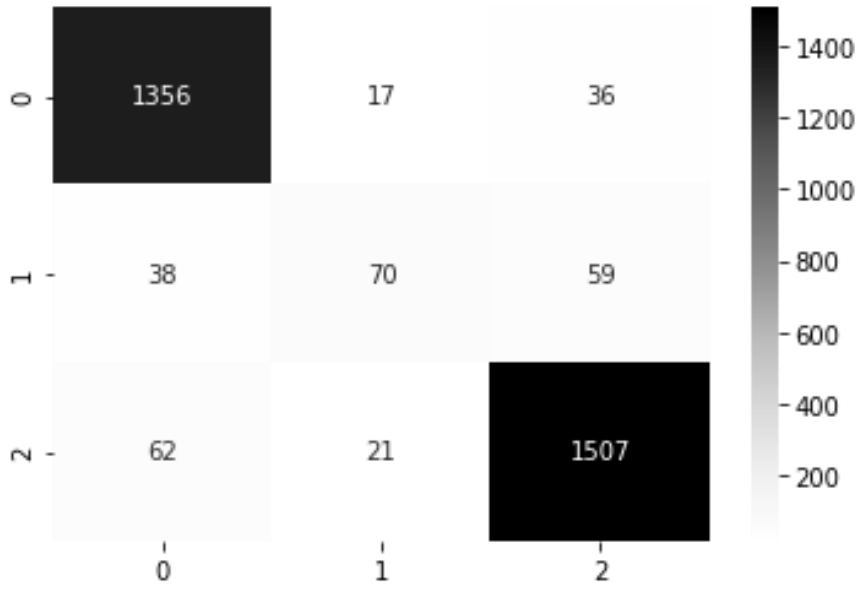
+ Nhãn Enjoyment có dữ liệu chiếm phân phối rất lớn so với các nhãn còn lại trên cả tập train và dev (1558/5548 chiếm 28,1% trên tập train và 214/545 chiếm 39,3% trên tập dev). Chính vì thế, hệ thống sẽ được học nhiều hơn, dẫn đến kết quả dự đoán chính xác cao.

+ Tuy nhiên, nhãn Fear thì lại khác. Nhãn Fear có dữ liệu chiếm phân phối khá nhỏ so với các nhãn còn lại (318/5548 chiếm 5,7% trên tập train và 31/545 chiếm 5,7% trên tập dev) nhưng vẫn đạt được độ chính xác cao. Trường hợp này, có thể do nhãn Fear dễ nhận biết thông qua các yếu tố từ ngữ như: sợ, huhu, sợ quá, ... hoặc các từ quá đặc trưng cho nhãn Fear.

c) UIT-VSFC [2]

– Nhãn **Sentiment**.

+ Confusion matrix.



Hình 5.42: Confusion Matrix trên bộ UIT-VSFC trên nhãn Sentiment khi chạy bằng Bert4News

- Ta thấy nhãn 0 và 2 có số lượng nhãn dự đoán đúng rất cao (Nhãn 0: 1356/1409 chiếm 96,2% và nhãn 2: 1507/1590 chiếm 94,8%).
- Trong khi đó, nhãn 1 thì khá thấp (70/167 chiếm 41,9%).

+ Kết quả đánh giá trên các độ đo.

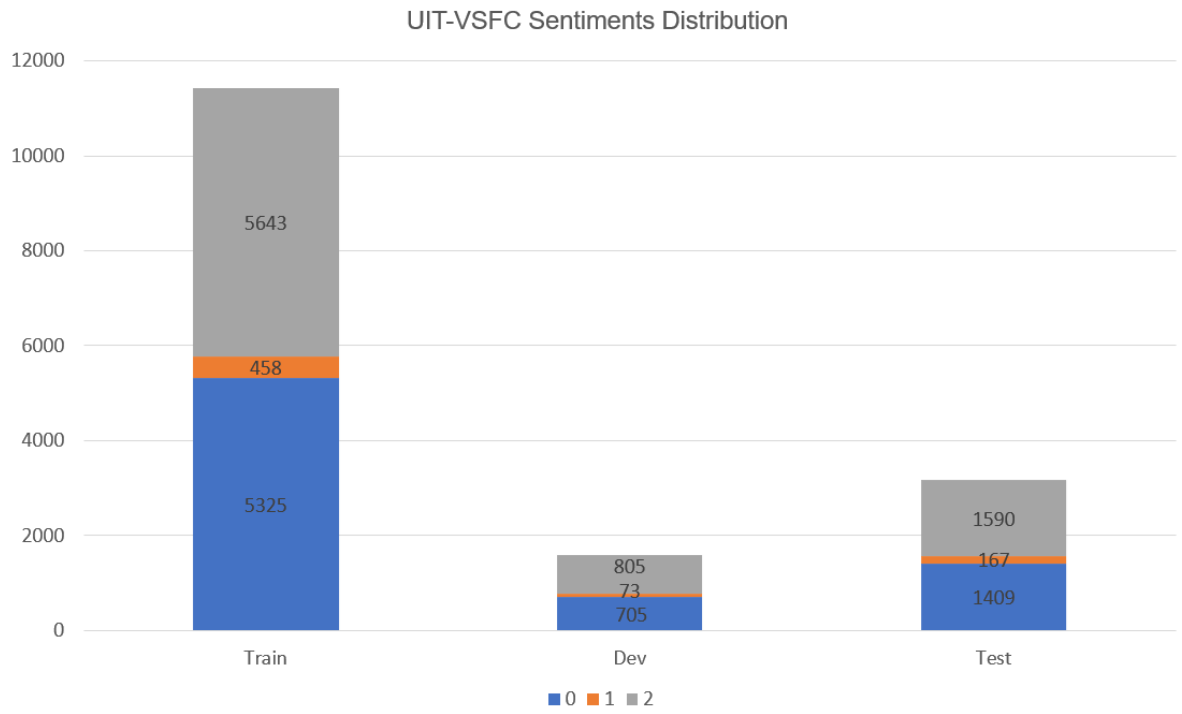
```
accuracy: 0.9264055590650663
F1 - macro: 0.7999744522340279
Classification report in Training set
              precision    recall  f1-score   support

           2       0.94      0.95      0.94      1590
           0       0.93      0.96      0.95      1409
           1       0.65      0.42      0.51       167

    accuracy                           0.93      3166
   macro avg       0.84      0.78      0.80      3166
weighted avg       0.92      0.93      0.92      3166
```

Hình 5.43: Kết quả đánh giá các độ đo trên bộ UIT-VSFC trên nhãn Sentiment khi chạy bằng Bert4News

Nhìn vào kết quả đánh giá ta thấy:

- Nhìn chung kết quả đánh giá trên các độ đo khá cao.
- Đặc biệt, Nhãn 0 và 2 có độ chính xác Precision rất cao (Nhãn 0: 0.93 và nhãn 2: 0.94). Tuy nhiên nhãn 1 lại có độ chính xác khá thấp so với 2 nhãn còn lại (0.65). Từ đó, làm cho độ chính xác bị giảm xuống (accuracy đạt 0.93).

+ Về lý do 2 nhãn 0 và 2 này đạt được kết quả cao nhưng nhãn 1 lại khá thấp như vậy, ta có thể xem qua phân phối các nhãn ở hình dưới:

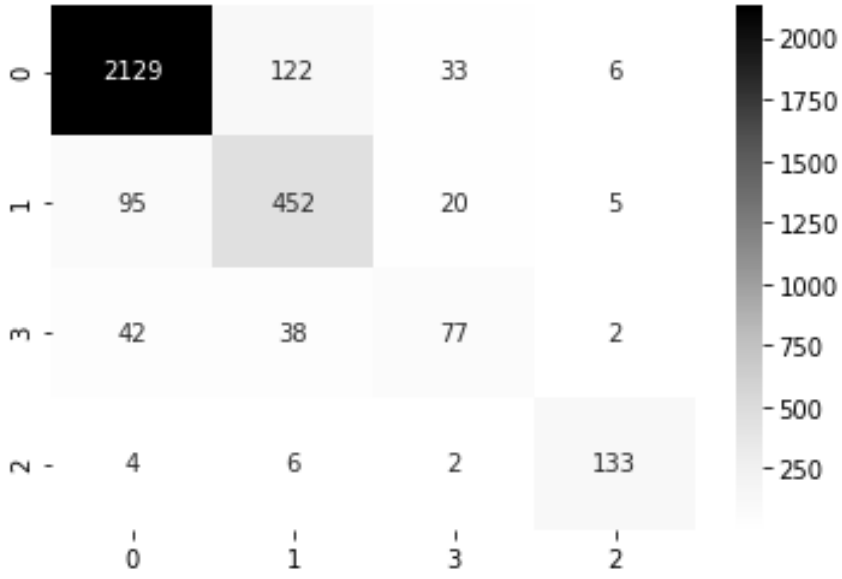


Hình 5.44: Phân phối các nhãn của bộ dữ liệu UIT-VSFC trên nhãn Sentiment

- Nhãn 0 và 2 có dữ liệu chiếm phân phối rất lớn so với các nhãn còn lại trên cả tập train và dev (Nhãn 0: 5325/11426 chiếm 46,6% trên tập train và 705/1583 chiếm 44,5% trên tập dev; Nhãn 2: 5643/11426 chiếm 23,1% trên tập train và 805/1583 chiếm 50,9% trên tập dev). Chính vì thế, hệ thống sẽ được học nhiều hơn, dẫn đến kết quả dự đoán chính xác cao.
- Tuy nhiên, nhãn 1 thì lại khác. Nhãn 1 có dữ liệu chiếm phân phối khá nhỏ so với các nhãn còn lại (458/11426 chiếm 4,0% trên tập train và 73/1583 chiếm 4,6% trên tập dev). Vì vậy, hệ thống sẽ được học ít hơn, dẫn đến kết quả dự đoán có thể ít chính xác hơn.

– Nhãn **Topic**.

+ Confusion matrix.



Hình 5.45: Confusion Matrix trên bộ UIT-VSFC trên nhãn Topic khi chạy bằng Bert4News

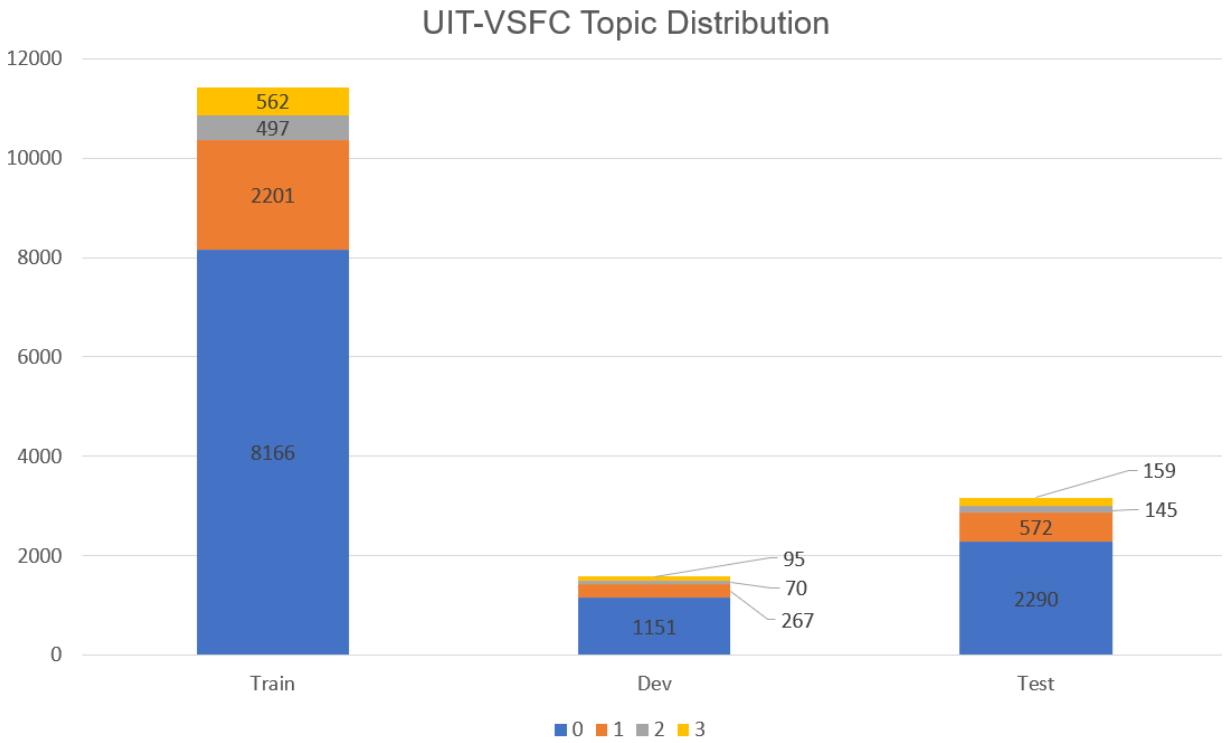
- Nhãn 0 có số lượng nhãn dự đoán chính xác rất cao (2129/2290 chiếm 95,6%). Tuy nhiên có đôi lúc nhãn 0 dễ nhận nhầm thành nhãn 1 (122/2290 chiếm 5,3%).
- Nhãn 3 dễ bị nhận nhầm sang nhãn khác (42 nhãn nhận nhầm thành nhãn 0 và 38 nhãn nhận nhầm thành nhãn 1), trong khi đó số nhãn 3 được dự đoán chính xác đạt 77 nhãn.

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.8815540113708149
F1 - macro: 0.7841836911107362
Classification report in Training set
              precision    recall  f1-score   support

           0       0.94      0.93      0.93      2290
           1       0.73      0.79      0.76       572
           3       0.58      0.48      0.53       159
           2       0.91      0.92      0.91       145

    accuracy                           0.88      3166
   macro avg       0.79      0.78      0.78      3166
weighted avg       0.88      0.88      0.88      3166
```

Hình 5.46: Kết quả đánh giá các độ đo trên bộ UIT-VSFC trên nhãn Topic khi chạy bằng Bert4News

- Nhãn 3 có kết quả khá thấp so với các nhãn còn lại. Hai nhãn 0 và 2 đạt độ chính xác rất cao.



Hình 5.47: Phân phối các nhãn của bộ dữ liệu UIT-VSFC trên nhãn Topic

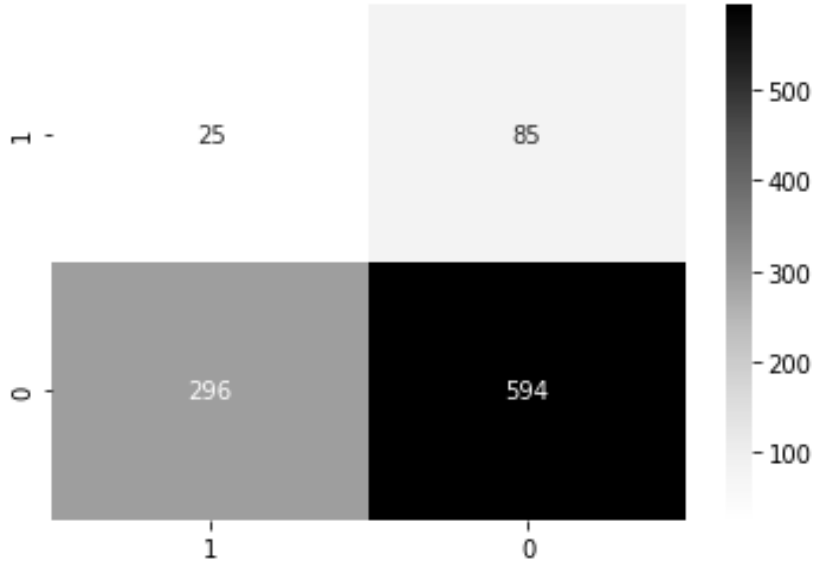
- Nhãn 0 có nhiều dữ liệu hơn so với các nhãn khác trên ccả 2 tập train và dev, còn nhãn số 3 thì ngược lại, chiếm số lượng rất nhỏ, không đáng kể.

Chính do vậy mà nhãn 0 đạt được kết quả chính xác cao do hệ thống được học nhiều còn ngược lại, nhãn 0 thì đạt kết quả khá thấp.

d) UIT-ViCTSD [3]

– Nhãn **Constructiveness**.

+ Confusion matrix.



Hình 5.48: Confusion Matrix trên bộ UIT-ViCTSD trên nhãn Constructiveness khi chạy bằng Bert4News

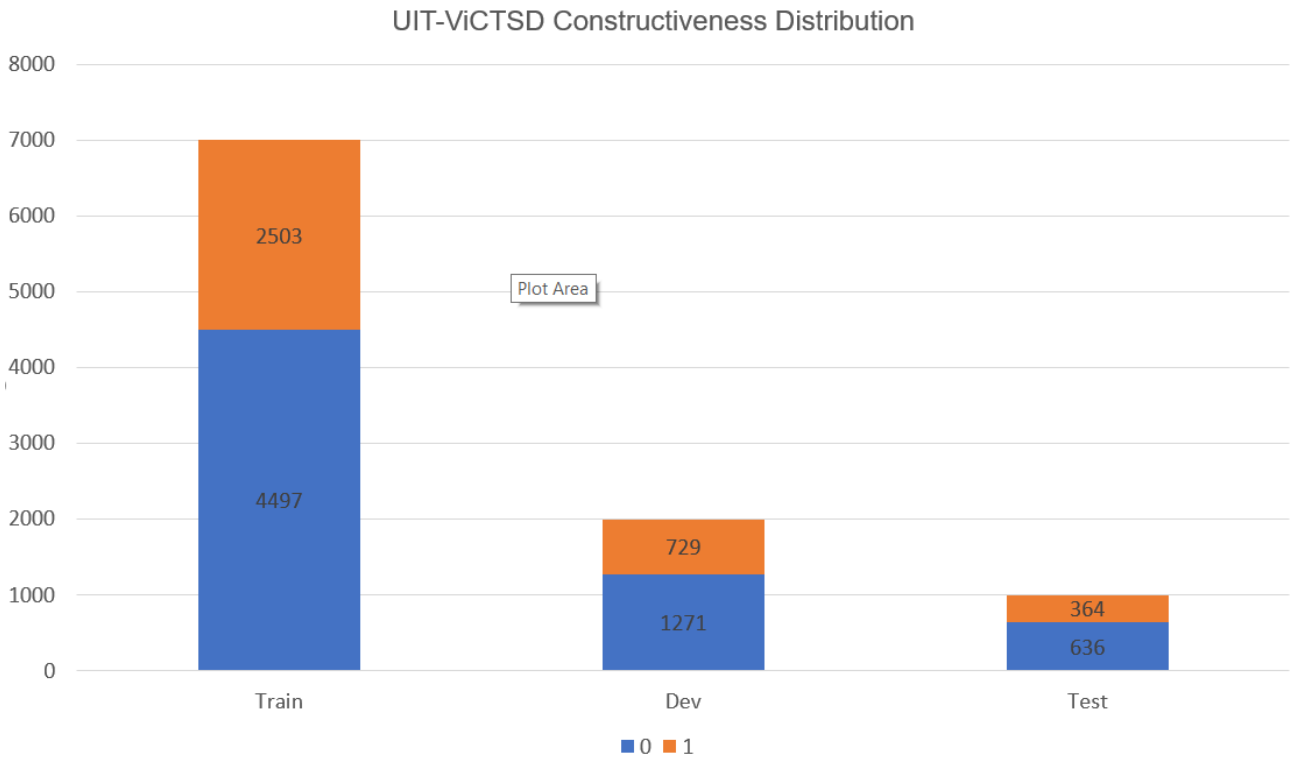
- Nhãn 0 rất dễ nhận nhầm thành nhãn 1 (296/594 nhãn bị nhận nhầm).

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.619
F1 - macro: 0.4365897264132947
Classification report in Training set
              precision    recall  f1-score   support

           1       0.08      0.23      0.12       110
           0       0.87      0.67      0.76       890

    accuracy                           0.62      1000
   macro avg       0.48      0.45      0.44      1000
weighted avg       0.79      0.62      0.69      1000
```

Hình 5.49: Kết quả đánh giá các độ đo trên bộ UIT-ViCTSD trên nhãn Constructiveness khi chạy bằng Bert4News

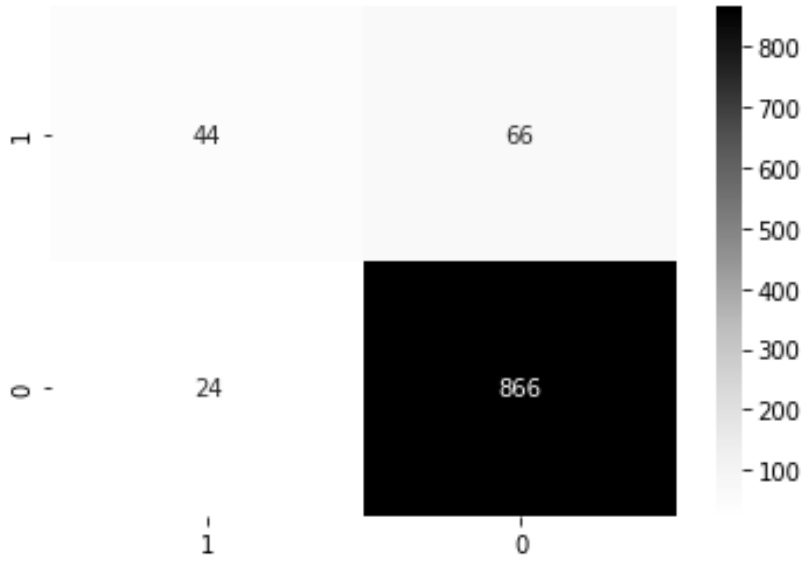
- Kết quả đánh giá trên nhãn 0 và 1 chênh lệch nhau rất nhiều. Nhãn 0 đạt độ chính xác 0.87 trong khi đó nhãn 1 chỉ được 0.08. Điều này làm cho accuracy bị kéo theo và chỉ đạt 0.62.
- Vì số lượng dữ liệu của 2 nhãn chênh lệch nhau rất nhiều như hình bên dưới:



Hình 5.50: Phân phối các nhãn của bộ dữ liệu UIT-ViCTSD trên nhãn Constructiveness

– Nhãn **Toxicity**.

+ Confusion matrix.



Hình 5.51: Confusion Matrix trên bộ UIT-ViCTSD trên nhãn Toxicity khi chạy bằng Bert4News

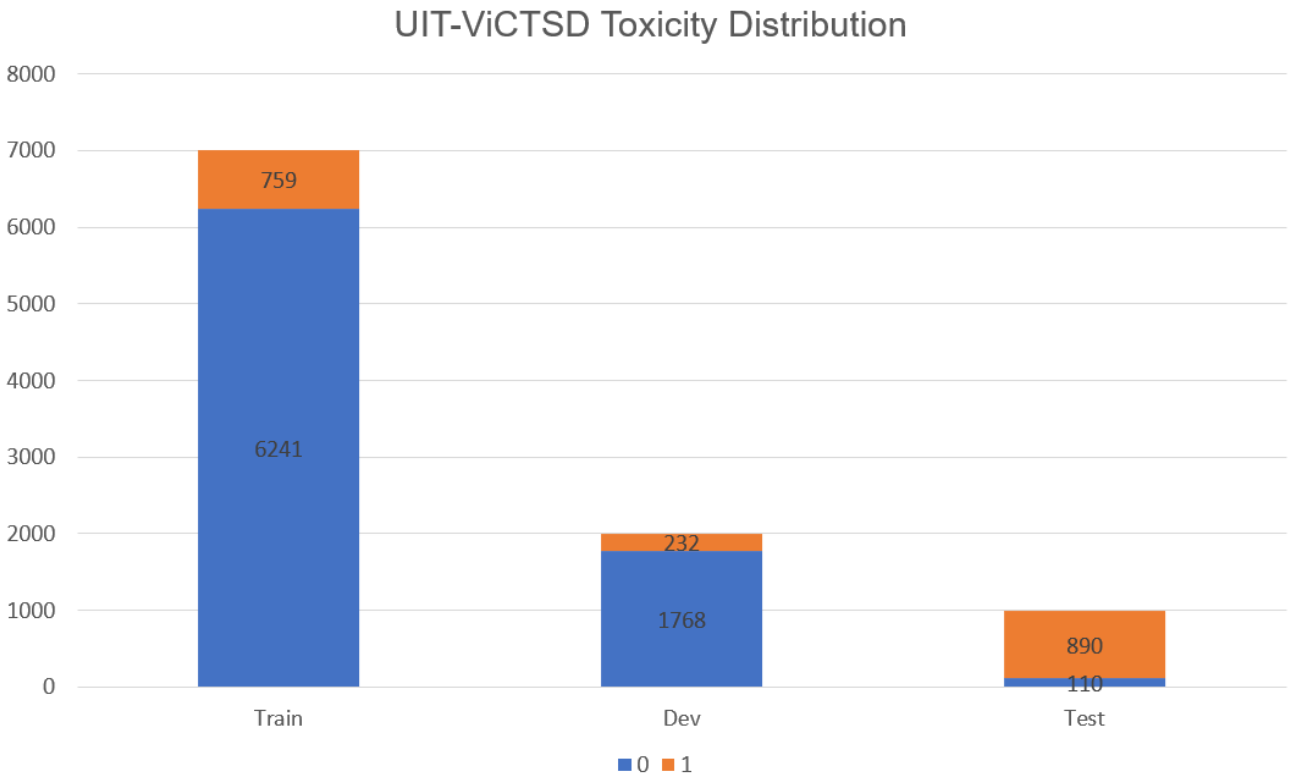
- Nhãn 0 có số lượng nhãn được dự đoán chính xác rất cao. Trong khi đó nhãn 1 thì lại khá thấp mà bị nhận nhầm sang nhãn 0 (66/110 nhãn bị nhận nhầm).

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.91
F1 - macro: 0.7224928773171845
Classification report in Training set
              precision    recall  f1-score   support

           1       0.65      0.40      0.49       110
           0       0.93      0.97      0.95       890

    accuracy                           0.91      1000
   macro avg       0.79      0.69      0.72      1000
weighted avg       0.90      0.91      0.90      1000
```

Hình 5.52: Kết quả đánh giá các độ đo trên bộ UIT-ViCTSD trên nhãn Toxicity khi chạy bằng Bert4News

- Nhãn 0 đạt kết quả cao hơn nhãn 1 vì phân phối dữ liệu của 2 nhãn trên tập dữ liệu không đồng đều (Nhãn 0 nhiều hơn nhãn 1). Ta có thể thấy chi tiết ở hình bên dưới:



Hình 5.53: Phân phối các nhãn của bộ dữ liệu UIT-ViCTSD trên nhãn Toxicity

5.2.5 XLM-R

a) Quy trình huấn luyện dữ liệu

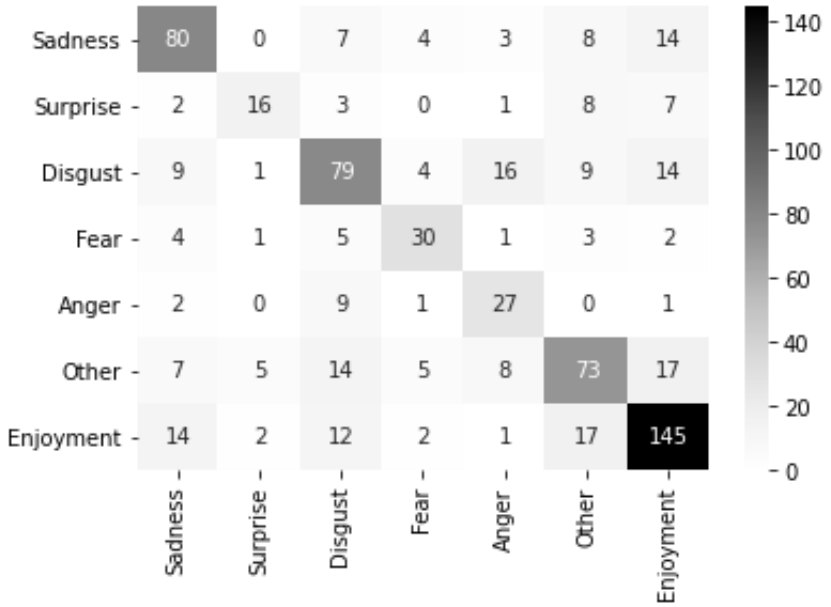
Tương tự như 2 phương pháp PhoBERT [7] (được trình bày ở mục 5.2.3) và Bert4News [5] [8] (được trình bày ở mục 5.2.4) thì XLM-R [10] cũng được train tương tự, ta có bảng thống kê sau.

Bộ dữ liệu	Label	Lần train	Loss thấp nhất	Checkpoint	Epoch	Batch_size	Weight_Decay	Precision	Recall	Accuracy	F1 - macro
VSMEC	Emotion	**1**	**0.0005**	**15000**	**100**	**32**	**0.01**	**0.63**	**0.62**	**0.64935**	**0.62045**
		2	0.0008	17000	100	32	0.001	0.61	0.62	0.63348	0.61467
		3	0.0003	17000	100	32	0.0001	0.61	0.61	0.64069	0.60693
VSFC	Sentiment	**1**	**0.0001**	**25000**	**70**	**32**	**0.01**	**0.84**	**0.79**	**0.93051**	**0.80591**
		2	0.0002	22500	70	32	0.001	0.84	0.77	0.92672	0.79902
		2	0.0001	24500	70	32	0.0001	0.83	0.76	0.92514	0.78743
	Topics	1	0.0001	33000	100	32	0.01	0.8	0.79	0.87997	0.79391
		2	0.0001	24000	70	32	0.001	0.8	0.76	0.88029	0.781
		3	**0.0001**	**24500**	**70**	**32**	**0.0001**	**0.8**	**0.78**	**0.88345**	**0.78954**
ViCTSD	Constructiveness	**1**	**0.0007**	**19500**	**100**	**32**	**0.01**	**0.47**	**0.43**	**0.568**	**0.41614**
		2	0.0006	21000	100	32	0.001	0.48	0.45	0.554	0.41608
		3	0.0027	21000	100	32	0.0001	0.47	0.44	0.565	0.41609
	Toxicity	**1**	**0.0003**	**15500**	**100**	**32**	**0.01**	**0.75**	**0.68**	**0.901**	**0.70799**
		2	0.0003	21500	100	32	0.001	0.73	0.65	0.894	0.67963
		3	0.0002	18000	100	32	0.0001	0.75	0.67	0.901	0.70227

Hình 5.54: Quy trình huấn luyện dữ liệu bằng XLM-R

b) UIT-VSMEC [1]

– Confusion matrix.



Hình 5.55: Confusion Matrix trên bộ UIT-VSMEC khi chạy bằng XLM-R

+ Nhãn Enjoyment có số lượng nhãn được dự đoán đúng khá cao (145/200 đạt 72,50%).

+ Nhãn Sadness có số lượng nhãn được dự đoán đúng cũng khá cao (80/118 đạt 67.80%).

– Kết quả đánh giá trên các độ đo.

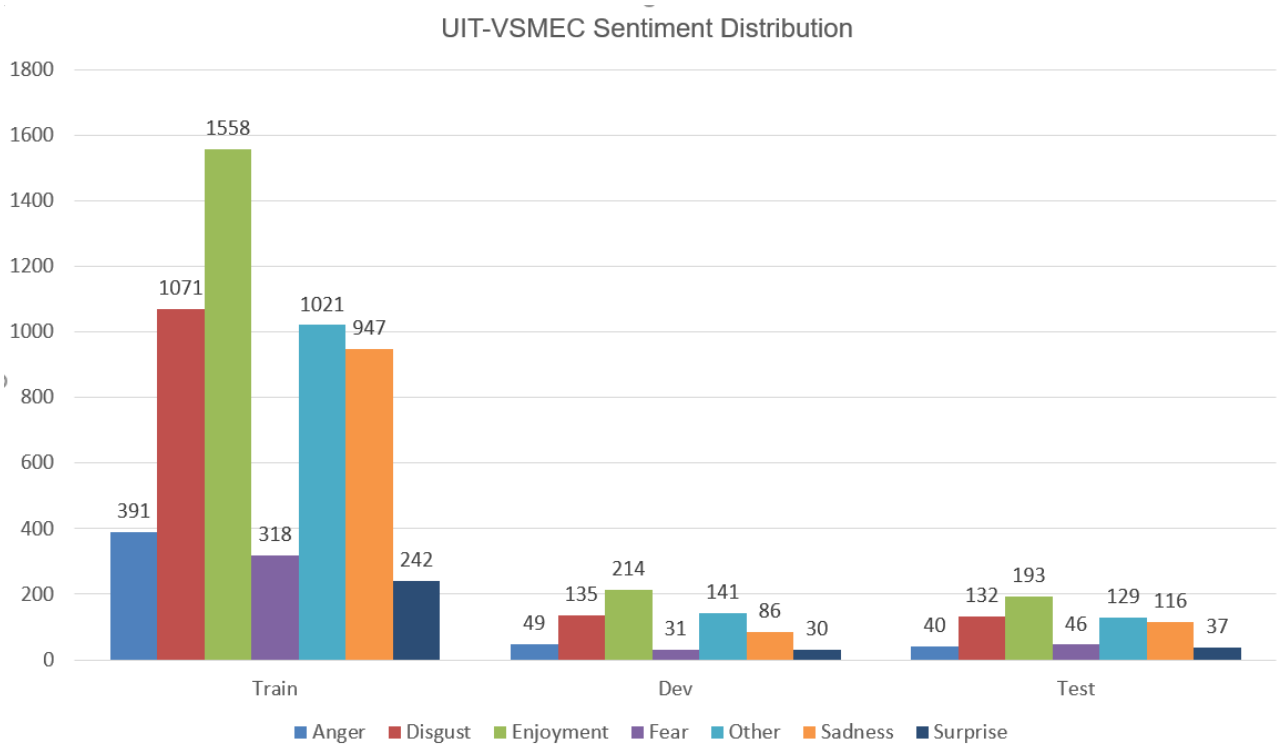
```
accuracy: 0.6493506493506493
F1 - macro: 0.6204478925826267
Classification report in Training set
              precision    recall  f1-score   support

     Sadness       0.68      0.69      0.68       116
    Surprise       0.64      0.43      0.52        37
     Disgust       0.61      0.60      0.61       132
        Fear       0.65      0.65      0.65        46
       Anger       0.47      0.68      0.56        40
       Other       0.62      0.57      0.59       129
   Enjoyment       0.72      0.75      0.74       193

    accuracy                           0.65       693
   macro avg       0.63      0.62      0.62       693
weighted avg       0.65      0.65      0.65       693
```

Hình 5.56: Kết quả đánh giá các độ đo trên bộ UIT-VSMEC khi chạy bằng XLM-R

+ Nhìn chung kết quả đánh giá trên các độ đo khá thấp và khá đồng đều trên các nhãn.

+ Đặc biệt, nhãn Enjoyment và Sadness có độ chính xác Precision rất cao (Enjoyment 0.72 và Sadness 0.68).

Về lý do 2 nhãn này đạt được kết quả cao như vậy, ta có thể xem qua phân phối các nhãn ở hình dưới:



Hình 5.57: Phân phối các nhãn của bộ dữ liệu UIT-VSMEC

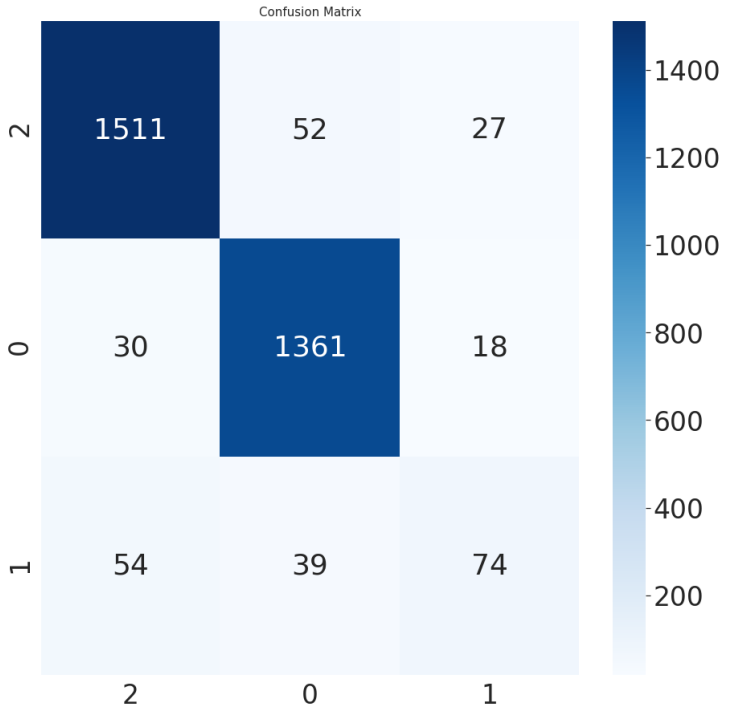
+ Nhãn Enjoyment có dữ liệu chiếm phân phối rất lớn so với các nhãn còn lại trên cả tập train và dev (1558/5548 chiếm 28,1% trên tập train và 214/545 chiếm 39,3% trên tập dev). Chính vì thế, hệ thống sẽ được học nhiều hơn, dẫn đến kết quả dự đoán chính xác cao.

+ Bên cạnh đó, nhãn Sadness cũng tương tự. Nhãn Sadness có dữ liệu chiếm phân phối khá cao so với các nhãn còn lại (947/5548 chiếm 17.07% trên tập train và 86/545 chiếm 15.78% trên tập dev) nên đạt được độ chính xác cao.

c) UIT-VSFC [2]

– Nhãn **Sentiment**.

+ Confusion matrix.



Hình 5.58: Confusion Matrix trên bộ UIT-VSFC trên nhãn Sentiment khi chạy bằng XLM-R

- Ta nhận thấy rằng nhãn 0 và 2 có số lượng nhãn dự đoán đúng rất cao (Nhãn 0: 1361/1452 chiếm 93.73% và nhãn 2: 1511/1595 chiếm 94.73%)
- Bên cạnh đó, nhãn số 1 dễ bị nhận nhầm sang nhãn 0 và 2 với 93/167 tổng số nhãn chiếm tới 55.69%.

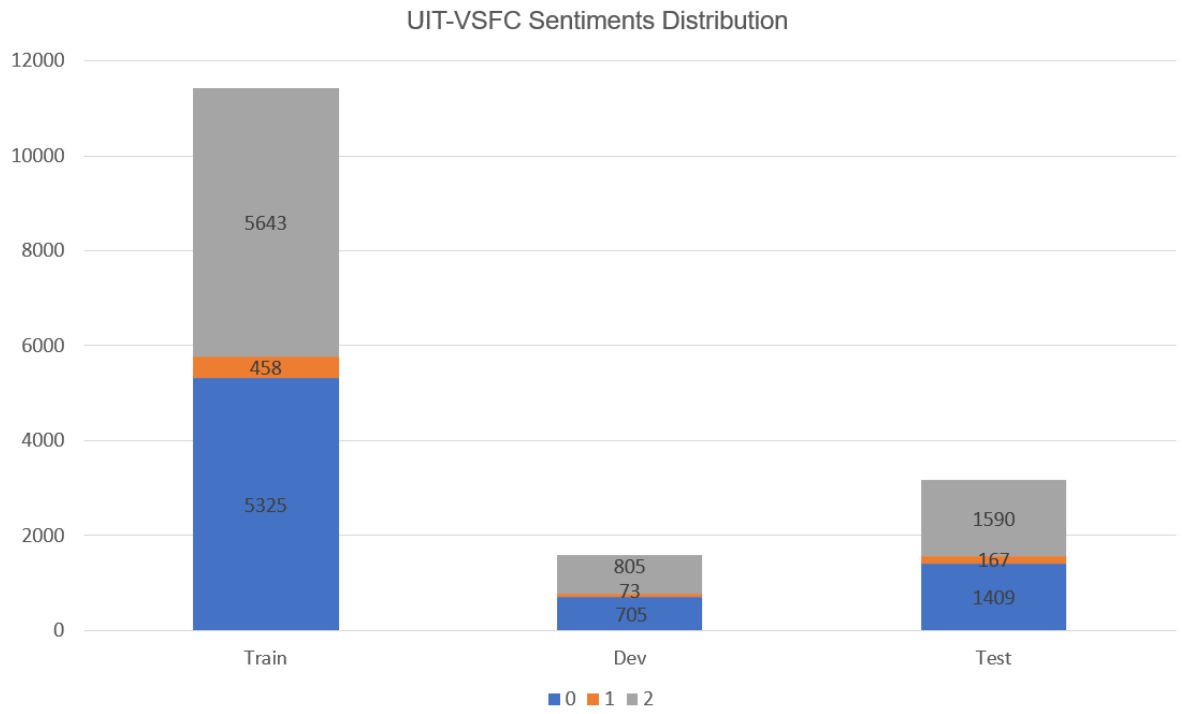
+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.930511686670878
F1 - macro: 0.8059069041342971
Classification report in Training set
              precision    recall  f1-score   support

           2       0.95      0.95      0.95      1590
           0       0.94      0.97      0.95      1409
           1       0.62      0.44      0.52       167

    accuracy                           0.93      3166
   macro avg       0.84      0.79      0.81      3166
weighted avg       0.93      0.93      0.93      3166
```

Hình 5.59: Kết quả đánh giá các độ đo trên bộ UIT-VSFC trên nhãn Sentiment khi chạy bằng XLM-R

- Nhìn chung kết quả đánh giá trên các độ đo khá cao.
- Đặc biệt, Nhãn 0 và 2 có độ chính xác Precision rất cao (Nhãn 0: 0.94 và nhãn 2: 0.95). Tuy nhiên nhãn 1 lại có độ chính xác khá thấp so với 2 nhãn còn lại (0.62). Từ đó, làm cho độ chính xác bị giảm xuống (accuracy đạt 0.93).

+ Về lý do 2 nhãn 0 và 2 này đạt được kết quả cao nhưng nhãn 1 lại khá thấp như vậy, ta có thể xem qua phân phối các nhãn ở hình dưới:

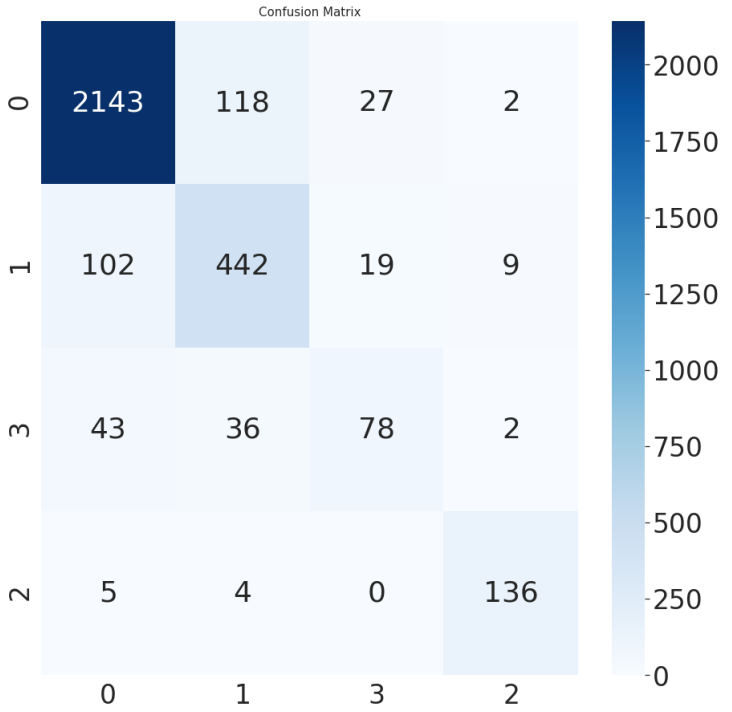


Hình 5.60: Phân phối các nhãn của bộ dữ liệu UIT-VSFC trên nhãn Sentiment

- Nhãn 0 và 2 có dữ liệu chiếm phân phối rất lớn so với các nhãn còn lại trên cả tập train và dev (Nhãn 0: 5325/11426 chiếm 46,6% trên tập train và 705/1583 chiếm 44,5% trên tập dev; Nhãn 2: 5643/11426 chiếm 23,1% trên tập train và 805/1583 chiếm 50,9% trên tập dev). Chính vì thế, hệ thống sẽ được học nhiều hơn, dẫn đến kết quả dự đoán chính xác cao.
- Tuy nhiên, nhãn 1 thì lại khác. Nhãn 1 có dữ liệu chiếm phân phối khá nhỏ so với các nhãn còn lại (458/11426 chiếm 4,0% trên tập train và 73/1583 chiếm 4,6% trên tập dev). Vì vậy, hệ thống sẽ được học ít hơn, dẫn đến kết quả dự đoán có thể ít chính xác hơn.

– Nhãn **Topic**.

+ Confusion matrix.



Hình 5.61: Confusion Matrix trên bộ UIT-VSFC trên nhãn Topic khi chạy bằng XLM-R

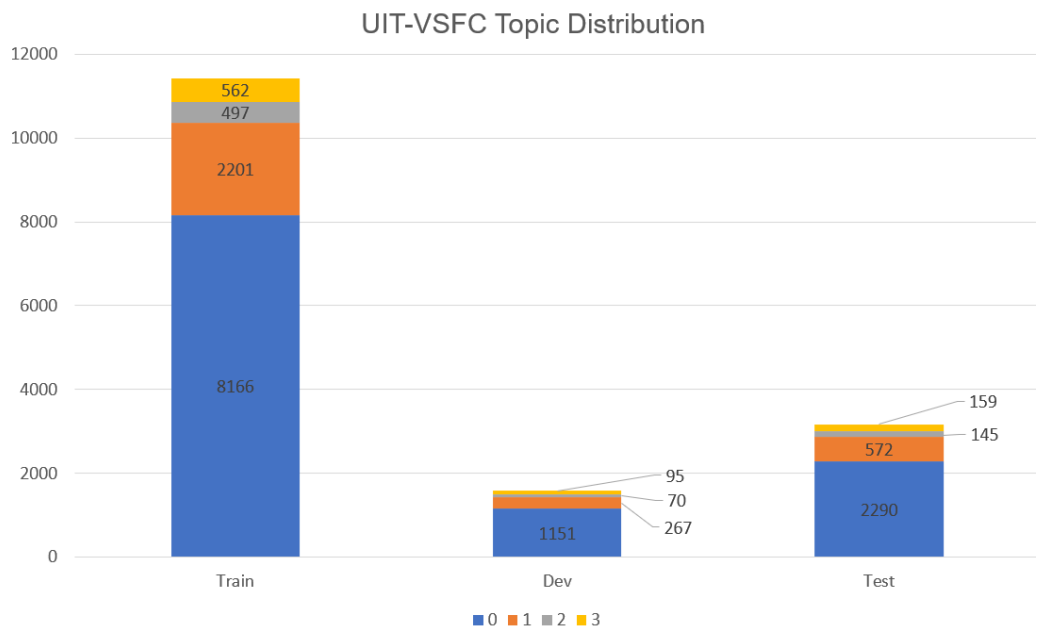
- Nhãn 0 có số lượng nhãn nhận chính xác rất cao (2143/2290 chiếm 93,58%). Tuy nhiên có đôi lúc nhãn 0 dễ nhận nhầm thành nhãn 1 (118/2290 chiếm 5,15%).
- Nhãn 3 dễ bị nhận nhầm sang nhãn khác (43 nhãn nhận nhầm thành nhãn 0 và 36 nhãn nhận nhầm thành nhãn 1), trong khi đó số nhãn 3 được dự đoán chính xác đạt 78 nhãn.

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.8834491471888819
F1 - macro: 0.7895410460826123
Classification report in Training set
              precision    recall  f1-score   support

           0       0.94      0.93      0.94      2290
           1       0.74      0.77      0.75       572
           3       0.62      0.49      0.55       159
           2       0.91      0.94      0.92       145

    accuracy                           0.88      3166
   macro avg       0.80      0.78      0.79      3166
weighted avg       0.88      0.88      0.88      3166
```

Hình 5.62: Kết quả đánh giá các độ đo trên bộ UIT-VSFC trên nhãn Topic khi chạy bằng XLM-R

- Nhãn 3 có kết quả khá thấp so với các nhãn còn lại. Hai nhãn 0 và 2 đạt độ chính xác rất cao.



Hình 5.63: Phân phối các nhãn của bộ dữ liệu UIT-VSFC trên nhãn Topic

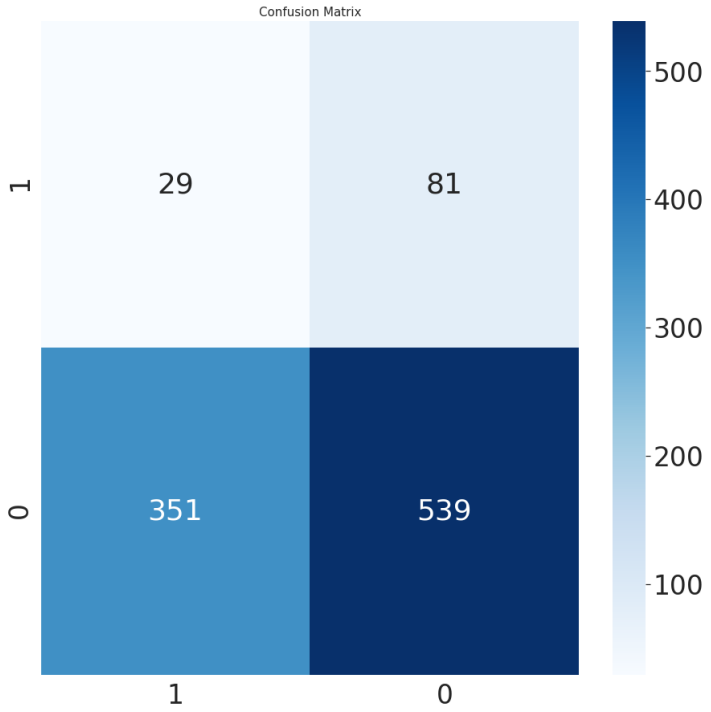
- Nhãn 0 có nhiều dữ liệu hơn so với các nhãn khác trên cả 2 tập train và dev, còn nhãn số 3 thì ngược lại, chiếm số lượng rất nhỏ, không đáng kể. Chính

do vậy mà nhãn 0 đạt được kết quả chính xác cao do hệ thống được học nhiều còn ngược lại, nhãn 3 thì đạt kết quả khá thấp.

d) UIT-ViCTSD [3]

– Nhãn **Constructiveness**.

+ Confusion matrix.



Hình 5.64: Confusion Matrix trên bộ UIT-ViCTSD trên nhãn Constructiveness khi chạy bằng XLM-R

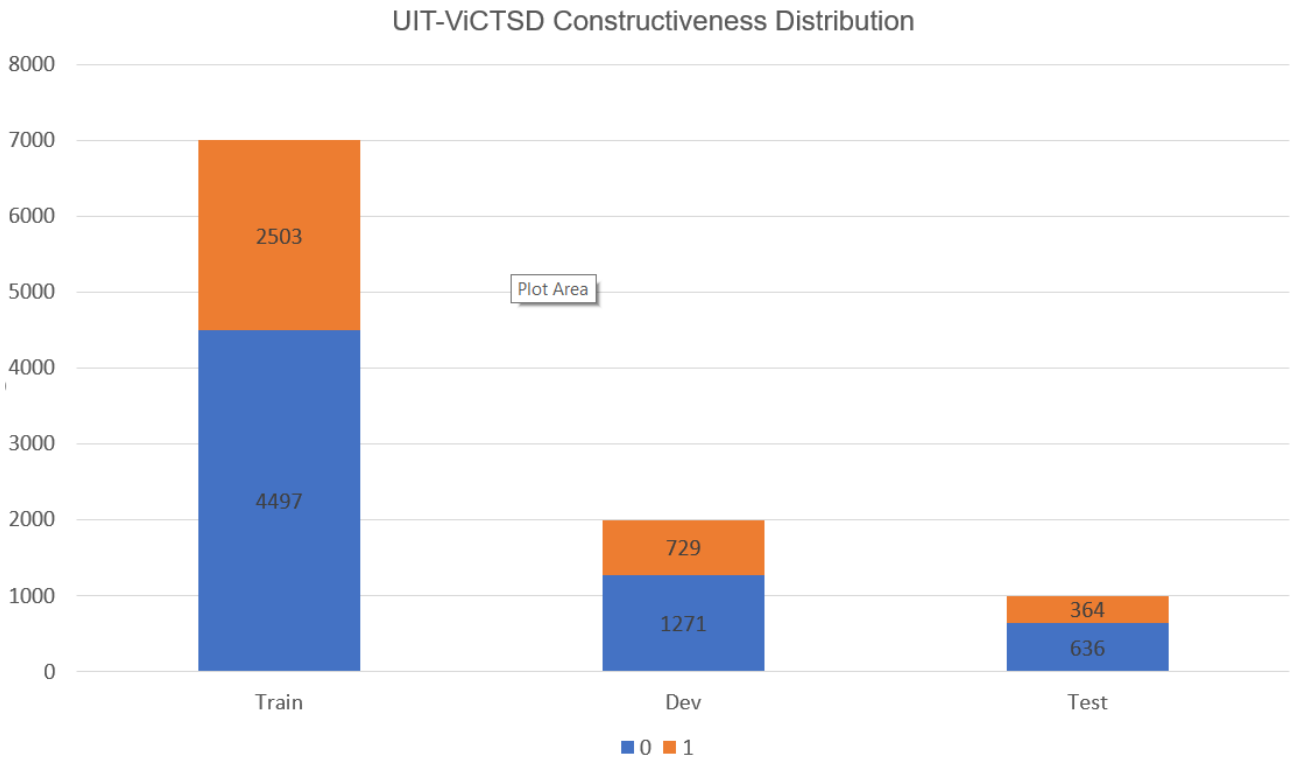
- Ta dễ dàng nhận thấy nhãn số 1 rất dễ bị dự đoán nhầm sang nhãn số 0 với 351/380 chiếm 92.37%.
- Bên cạnh đó, nhãn số 1 cũng dễ bị nhận nhầm sang nhãn số 0 với 81/110 chiếm 73.64%.

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.568
F1 - macro: 0.41613731585349373
Classification report in Training set
              precision    recall  f1-score   support

           1       0.08      0.26      0.12       110
           0       0.87      0.61      0.71       890

    accuracy                           0.57      1000
   macro avg       0.47      0.43      0.42      1000
weighted avg       0.78      0.57      0.65      1000
```

Hình 5.65: Kết quả đánh giá các độ đo trên bộ UIT-ViCTSD trên nhãn Constructiveness khi chạy bằng XLM-R

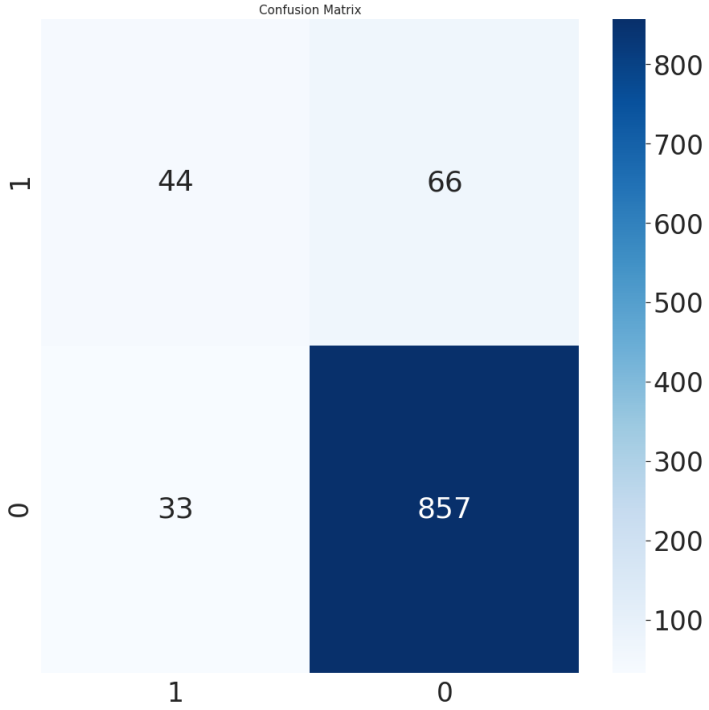
- Kết quả đánh giá trên nhãn 0 và 1 chênh lệch nhau rất nhiều. Nhãn 0 đạt độ chính xác 0.87 trong khi đó nhãn 1 chỉ được 0.08. Điều này làm cho accuracy bị kéo theo và chỉ đạt 0.57.
- Vì số lượng dữ liệu của 2 nhãn chênh lệch nhau rất nhiều như hình bên dưới:



Hình 5.66: Phân phối các nhãn của bộ dữ liệu UIT-ViCTSD trên nhãn Constructiveness

– Nhãn **Toxicity**.

+ Confusion matrix.



Hình 5.67: Confusion Matrix trên bộ UIT-ViCTSD trên nhãn Toxicity khi chạy bằng XLM-R

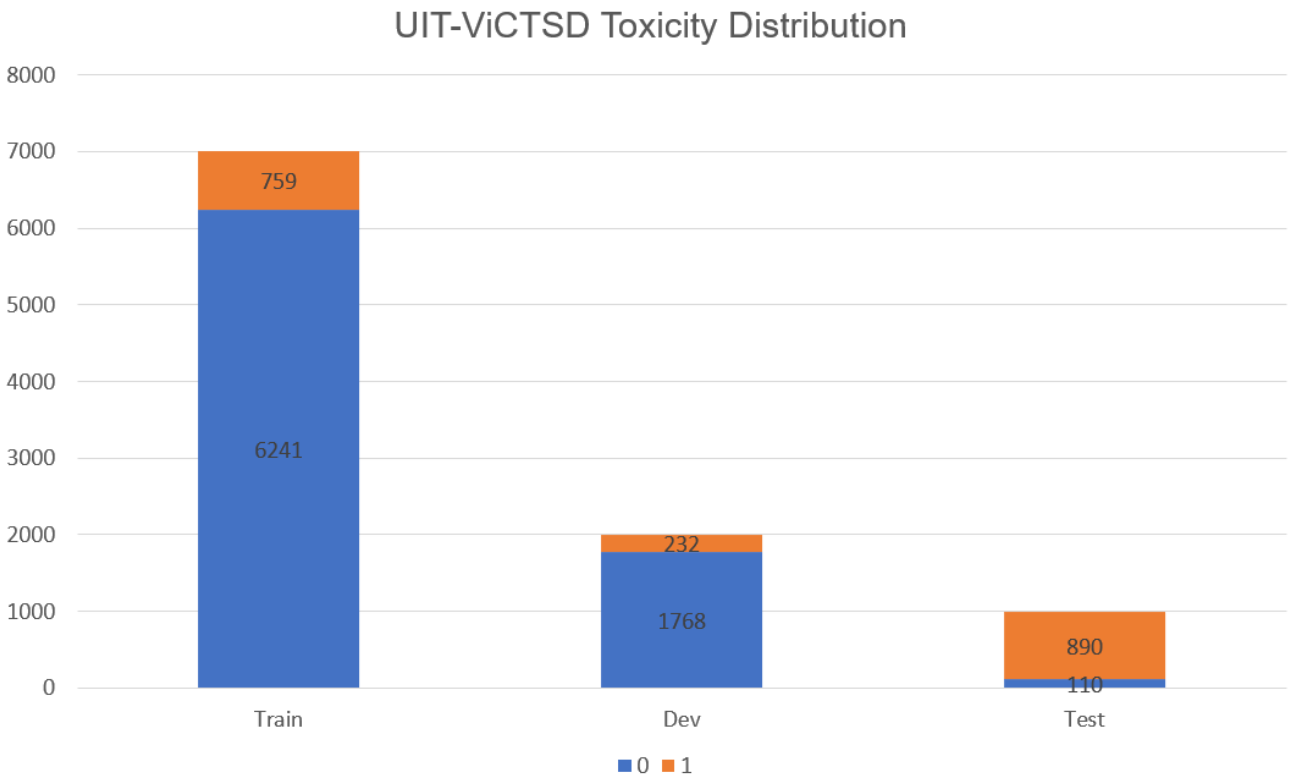
- Ở đây ta thấy nhãn số 1 dễ bị nhận nhầm sang nhãn số 0, số lượng nhận nhầm là 66/110 nhãn chiếm 60%.

+ Kết quả đánh giá trên các độ đo.

```
accuracy: 0.901
F1 - macro: 0.70799130462996
Classification report in Training set
              precision    recall  f1-score   support

           1       0.57      0.40      0.47       110
           0       0.93      0.96      0.95       890

    accuracy                           0.90      1000
   macro avg       0.75      0.68      0.71      1000
weighted avg       0.89      0.90      0.89      1000
```

Hình 5.68: Kết quả đánh giá các độ đo trên bộ UIT-ViCTSD trên nhãn Toxicity khi chạy bằng XLM-R

- Nhìn vào bảng đánh giá, ta thấy rằng độ chính xác của nhãn 1 thấp hơn nhãn số 0. Vậy nguyên nhân là vì sao?



Hình 5.69: Phân phối các nhãn của bộ dữ liệu UIT-ViCTSD trên nhãn Toxicity

- **Giải thích**: Để giải thích cho điều này, chúng ta hãy nhìn vào bảng phân phối của các nhãn trong nhãn **Toxicity**. Ta thấy trên tập train, số lượng nhãn số 1 ít hơn rất nhiều so với nhãn số 0, dẫn đến dữ liệu sẽ không học được đủ nhiều cho nhãn số 1. Điều này dẫn đến nhãn số 1 có độ chính xác thấp hơn.

Chương 6 PHÂN TÍCH KẾT QUẢ THỰC NGHIỆM

6.1 Bộ dữ liệu UIT-VSMEC

Algorithm	Accuracy(%)	F1-score(%)
RandomForest + BoW	50.64	40.11
SVM+BoW	58.00	56.87
LSTM+word2Vec	53.39	53.30
LSTM+fastText	54.25	53.77
CNN+word2Vec	59.74	59.74
CNN+fastText	56.85	56.79
Logistic Regression	48.63	42.92
CNN-LSTM	55.99	55.04
PhoBERT	62.77	61.07
BERT4NEWS	61.61	59.71
XLM-R	64.94	62.04

Bảng 2: Bảng so sánh các độ đo giữa các phương pháp có trong bài báo [1] và các phương pháp chạy thực nghiệm trên bộ dữ liệu UIT-VSMEC

- Ở đây, 6 phương pháp đầu là có trong bài báo, 5 phương pháp sau là chúng tôi chạy thực nghiệm.
- Sau khi chạy thực nghiệm thì phương pháp **Logistic Regression** có độ chính xác thấp nhất với 48.63%. Vì đây là phương pháp truyền thống nên nó chưa đủ độ phức tạp để xử lý bài toán này.
- Trong khi đó, phương pháp **XLM-R** cho độ chính xác cao nhất với 64.94%. Vì đây là phương pháp mới (SOTA) nên nó đủ độ phức tạp để xử lý bài toán này.
- Trong số 6 phương pháp được thực nghiệm trong bài báo, phương pháp CNN+word2Vec có độ chính xác cao nhất với accuracy và F1 cùng bằng 59.74% trong khi phương pháp XLM-R của chúng tôi đạt độ chính xác cao nhất là 64.94%

đối với accuracy và 62.04% trên F1. Độ chính xác cải thiện 5.20% đối với accuracy và 2.30% đối với F1.

6.2 Bộ dữ liệu UIT-VSFC

		Sentiment			Topic		
Algorithm	Feature	Precision	Recall	F1	Precision	Recall	F1
NB	Uni-gram	86.10	84.60	85.30	79.90	84.60	81.20
	Bi-gram	88.30	86.80	87.50	84.20	86.80	85.40
	Bi-gram + DEP	88.20	86.80	87.50	84.40	86.80	85.50
	Bi-gram + DEP + POS	84.20	86.60	85.30	82.80	86.60	84.60
Maxent	Uni-gram	83.30	84.90	84.10	83.30	83.20	83.20
	Bi-gram	87.50	84.90	84.10	86.30	85.80	86.10
	Bi-gram + DEP	87.60	87.50	87.50	86.70	86.30	86.60
	Bi-gram + DEP + POS	87.40	87.30	87.30	86.60	85.90	86.20
LSTM	Word2Vec	88.40	87.10	87.60	88.20	87.20	87.70
Bi-LSTM	Word2Vec	90.80	93.40	92.00	89.30	90.00	89.60
Logistic Regression		74.00	58.00	58.00	80.00	64.00	68.00
CNN-LSTM		81.00	73.00	75.00	81.00	75.00	77.00
PhoBert		86.00	80.00	82.00	80.00	77.00	78.00
Bert4News		84.00	78.00	80.00	79.00	78.00	78.00
XLM-R		84.00	79.00	81.00	88.00	88.00	88.00

Bảng 3: Bảng so sánh các độ đo giữa các phương pháp có trong bài báo [13] và các phương pháp chạy thực nghiệm trên bộ dữ liệu UIT-VSFC

– Đối với nhãn **Sentiment**:

+ F1 của các phương pháp SOTA không có sự chênh lệch nhiều. Tuy nhiên, phương pháp **Logistic Regression** và **CNN-LSTM** có F1 thấp hơn các phương pháp còn lại.

+ Trong số 4 phương pháp được thực nghiệm trong bài báo, phương pháp Bi-LSTM+Word2Vec có độ chính xác cao nhất với F1 đạt 92.00% trong khi phương pháp PhoBert của chúng tôi đạt độ chính xác cao nhất với F1 là 82.00%. Độ chính xác giảm 10.00% đối với F1.

– Đối với nhãn **Topic**:

+ F1 của 3 phương pháp **CNN-LSTM**, **PhoBert**, **Bert4News** không có sự chênh lệch nhiều. Tuy nhiên, phương pháp **XLM-R** có F1 cao nhất với 88.00% và phương pháp **Logistic Regression** có F1 thấp nhất với 68.00%.

+ Trong số 4 phương pháp được thực nghiệm trong bài báo, phương pháp Bi-LSTM+Word2Vec có độ chính xác cao nhất với F1 đạt 89.60% trong khi

phương pháp XLM-R của chúng tôi đạt độ chính xác cao nhất với F1 là 88.00%. Độ chính xác giảm 1.60% đối với F1.

6.3 Bộ dữ liệu UIT-ViCTSD

	Constructiveness		Toxicity	
Algorithm	Accuracy(%)	F1-score(%)	Accuracy(%)	F1-score(%)
Logistic Regression	79.91	70.78	90.27	55.35
Random Forest	79.10	73.75	90.03	90.03
SVM	78.00	76.10	90.17	59.06
LSTM + fastText	80.00	76.26	88.90	49.63
LSTM + PhoW2V	78.20	77.42	89.00	49.70
LSTM + PhoW2V	79.90	77.53	89.10	48.88
LSTM + PhoW2V	79.50	77.94	88.90	49.62
Our system	79.40	78.59	88.42	59.40
Logistic Regression	75.90	71.17	89.60	52.41
CNN-LSTM	80.60	79.46	89.30	59.71
PhoBert	54.60	41.01	90.60	72.92
Bert4News	61.90	43.66	91.00	72.25
XLM-R	56.80	41.61	90.10	70.80

Bảng 4: Bảng so sánh các độ đo giữa các phương pháp có trong bài báo [3] và các phương pháp chạy thực nghiệm trên bộ dữ liệu UIT-ViCTSD

– Đối với nhãn **Constructiveness**:

+ Độ chính xác của các phương pháp SOTA thấp hơn so với phương pháp **Logistic Regression** và **CNN-LSTM**.

+ Trong số 8 phương pháp được thực nghiệm trong bài báo, phương pháp LSTM+fastText có độ chính xác cao nhất với accuracy là 80.00% trong khi phương pháp CNN-LSTM của chúng tôi đạt độ chính xác cao nhất là 80.60% đối với accuracy. Độ chính xác cải thiện 0.60% đối với accuracy.

+ Phương pháp Our system có độ chính xác cao nhất với F1 là 78.59% trong khi phương pháp CNN-LSTM của chúng tôi đạt độ chính xác cao nhất là 79.46% đối với F1. Độ chính xác cải thiện 0.87% đối với F1.

- Đối với nhãn **Toxicity**:
  + Độ chính xác của các phương pháp hầu như không có sự chênh lệch nhiều.
  + Trong số 8 phương pháp được thực nghiệm trong bài báo, phương pháp Logistic Regression có độ chính xác cao nhất với accuracy là 90.27% trong khi phương pháp CNN-LSTM của chúng tôi đạt độ chính xác cao nhất là 91.00% đối với accuracy. Độ chính xác cải thiện 0.73% đối với accuracy.
  + Phương pháp Random Forest có độ chính xác cao nhất với F1 là 90.03% trong khi phương pháp PhoBert của chúng tôi đạt độ chính xác cao nhất là 72.92% đối với F1. Độ chính xác giảm 17.11% đối với F1.

Chương 7 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

7.1 Kết luận

Ở đồ án này, nhóm tiến hành phân loại văn bản Tiếng Việt trên năm mô hình bao gồm Logistic Regression, CNN-LSTM [4], PhoBert [7], Bert4News [5] [8], XLM-R [10] trên ba bộ dữ liệu UIT-VSMEC [1], UIT-VSFC [2], UIT-ViCTSD [3]. Sau khi so sánh kết quả nhóm rút ra được kết luận rằng: nhìn chung độ chính xác trên các phương pháp thực nghiệm vẫn chưa cao, phương pháp Logistic Regression có độ chính xác khá thấp, các phương pháp SOTA có độ chính xác khá cao. Các phương pháp tiền xử lý như xóa stopword, xóa các ký tự đặc biệt, tách từ,… chưa phát huy được công dụng nhiều dẫn đến độ chính xác chưa được cải thiện nhiều. Phân bố dữ liệu trên các nhãn của các bộ ngữ liệu không cân bằng, nhiều nhãn có dữ liệu quá ít dẫn đến kết quả dự đoán sai.

7.2 Hướng phát triển

Trong tương lai nhóm mong muốn sẽ cải thiện độ chính xác của các phương pháp. Nhóm sẽ nghiên cứu thêm phương pháp xử lý dữ liệu tối ưu hơn với từng bộ ngữ liệu đặc trưng, tiến hành thực nghiệm trên các mô hình SOTA mới hơn. Ngoài ra do sự phân bổ của kho dữ liệu nên nhóm sẽ thu thập thêm dữ liệu để tăng cường cho các lớp ít dữ liệu.

TÀI LIỆU THAM KHẢO

[1] "HO, Vong Anh, et al. Emotion recognition for vietnamese social media text. In: International Conference of the Pacific Association for Computational Linguistics. Springer, Singapore, 2019. p. 319-333.".

[2] "VAN NGUYEN, Kiet, et al. Uit-vsfc: Vietnamese students' feedback corpus for sentiment analysis. In: 2018 10th International Conference on Knowledge and Systems Engineering (KSE). IEEE, 2018. p. 19-24.".

[3] "NGUYEN, Luan Thanh; VAN NGUYEN, Kiet; NGUYEN, Ngan Luu-Thuy. Constructive and toxic speech detection for open-domain social media comments in vietnamese. arXiv preprint arXiv:2103.10069, 2021.".

[4] "KIPYATKOVA, Irina; KARPOV, Alexey. Class-based LSTM Russian language model with linguistic information. In: Proceedings of The 12th Language Resources and Evaluation Conference. 2020. p. 2470-2474.".

[5] "VAN THIN, Dang, et al. Investigating Monolingual and Multilingual BERTModels for Vietnamese Aspect Category Detection. arXiv preprint arXiv:2103.09519, 2021.".

[6] "LIU, Yinhan, et al. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692, 2019.".

[7] "NGUYEN, Dat Quoc; NGUYEN, Anh Tuan. PhoBERT: Pre-trained language models for Vietnamese. arXiv preprint arXiv:2003.00744, 2020.".

[8] "NGUYEN, Thanh Chinh; NGUYEN, Van Nha. NLPBK at VLSP-2020 shared task: Compose transformer pretrained models for Reliable Intelligence Identification on Social network. arXiv preprint arXiv:2101.12672, 2021.".

[9] “LAMPLE, Guillaume; CONNEAU, Alexis. Cross-lingual language model pretraining. arXiv preprint arXiv:1901.07291, 2019.”.

[10] “CONNEAU, Alexis, et al. Unsupervised cross-lingual representation learning at scale. arXiv preprint arXiv:1911.02116, 2019.”.

[11] "Python Vietnamese Toolkit," [Online]. Available: https://pypi.org/project/pyvi/. [Accessed 7 July 2021].

[12] D. Le, "vietnamese-stopwords," 7 September 2020. [Online]. Available: https://github.com/stopwords/vietnamese-stopwords. [Accessed 7 July 2021].

[13] “Nguyen, Phu XV, Tham TT Hong, Kiet Van Nguyen, and Ngan Luu-Thuy Nguyen. "Deep learning versus traditional classifiers on Vietnamese students’ feedback corpus." In 2018 5th NAFOSTED Conference on Information and Computer Science (NICS), pp. 75-80. IEEE, 2”.