Áp dụng phương pháp Ensemble PhoBERT với FastText trong tác vụ phân tích trạng thái

Đặng Anh Tiến
20520800@gm.uit.edu.vn

Nguyễn Vinh Tiệp
tiepnv@uit.edu.vn

***Tóm tắt nội dung*—Phân tích trạng thái (Sentiment Analysis) là bài toán quan trọng trong xử lý ngôn ngữ tự nhiên, tác vụ hỗ trợ rất tốt trong việc hiểu được các thông tin phản hồi từ người dùng. Mô hình Text Classification thường được áp dụng để giải quyết bài toán này, sử dụng các thuật toán Machine Learning cũng đạt được kết quả có thể chấp được. Tuy nhiên, lĩnh vực Deep Learning cho thấy sự vượt trội hơn với khả năng hiểu được thông tin ngữ cảnh. Tinh chỉnh BERT cho thấy độ chính xác tốt hơn đáng kể, ngoài ra áp dụng phương pháp Ensemble giúp cải thiện tình trạng overfitting.**

Từ khóa: PhoBERT, FastText, finetuning, ensemble.

I. Giới thiệu

Hiện nay, khách hàng thường nhìn nhận và để lại đánh giá đa dạng lĩnh vực trên các nền tảng trực tuyến. Các thông tin trải nghiệm người dùng này vô cùng quý giá, đó là nền tảng để khách hàng đưa ra các quyết định chi tiền, không khác một bản báo cáo chân thực đối với doanh nghiệp. Việc xử lý thủ công lượng lớn dữ liệu này vô cùng vất vả, vì vậy các hệ thống thực hiện Sentiment Analysis tự động xuất hiện, với khả năng học từ nguồn dữ liệu dồi dào, các mô hình có thể tự động thực hiện phân tích hàng triệu dòng feedback. Tác vụ đơn giản nhất của hệ thống là phân loại dữ liệu thuộc hai nhóm: **Tích cực (Positive)** cho biết sự hài lòng và **Tiêu cực (Negative)** biểu lộ sự không hài lòng của khách hàng.

Vào năm 2017, Transformer [1] ra đời và là một trong những nghiên cứu đột phá của lĩnh vực NLP vào thời điểm đó. Như tên bài báo ("Attention is all you need"), loại bỏ hoàn toàn kiến trúc recurent network cũ, chỉ còn các Multi Layer Perceptron kết nối với nhau cùng với cơ chế Self Attention cho phép học đồng thời các thông tin ngữ cảnh, giải quyết vấn đề long-term-dependency tồn đọng ở các kiến trúc cũ. Từ khi ra mắt, kiến trúc này trở nên ngày phổ biến cho bất kì bài toán NLP.

Tiếp nối sự thành công đó, BERT [2] nhanh chóng trở thành hiện tượng với cách tiếp cận khác với các mô hình ngôn ngữ trước. Thay vì sử dụng kiến trúc BiLSTM trích xuất thông tin ngữ cảnh từ hai hướng như ELMo [3]. Hay sử dụng Decoder của Transformer để trích xuất thông tin theo kiến trúc left-to-right như GPT [4]. BERT học đồng thời thông tin ngữ cảnh từ cả 2 chiều, sử dụng Encoder của Transformer giúp mô hình thành công trong việc tìm được các biểu diễn ngữ cảnh của từ hiệu quả hơn. Hiệu chỉnh BERT cho các tác vụ sau có độ chính xác cao.

Trong đồ án này, nhóm sẽ áp dụng BERT trên bộ dữ liệu Tiếng Việt, mục tiêu của đồ án:

1) Tinh chỉnh BERT trên bộ dữ liệu Tiếng Viêt.
2) So sánh kết quả của BERT với cách tiếp cận cũ sử dụng Word Embedding.
3) Áp dụng phương pháp Ensemble để tăng độ chính xác trên tập Test.

II. Các nghiên cứu liên quan

Sentiment Analysis là một bài toán nhỏ của tác vụ Text Classification, trong đó, với mỗi input sẽ được phân loại thuộc nhóm **Tích cực (Positive)** hay **Tiêu cực (Negative)**, một số bài toán sẽ có thêm nhóm **Trung tính (Neural)**.

Hướng tiếp cận hiệu chỉnh BERT trên bộ dữ liệu Tiếng Việt không còn mới. [5] tinh chỉnh PhoBERT [6] trong hai tác vụ (1) phân loại bình luận về lĩnh vực nhà hàng và ẩm thực trên Foody (2) bình luận của người dùng trên các sàn thương mại điện tử. Kết quả dụ đoán trên cả hai tác vụ này đều đạt độ chính xác cao, tốt nhất khi sử dụng PhoBERT-RCNN. [7] xây dựng hệ thống phân loại bình luận xúc phạm, ganh ghét trên bộ dữ liệu là các bình luận trên mạng xã hội. Kết quả tốt nhất khi sử dụng PhoBERT-CNN.

Ensemble là phương pháp phổ biến trong lĩnh vực Machine Learning, kết quả dự đoán là sự đóng góp của nhiều mô hình thành viên. Ensemble thường được biết đến là giải pháp không những khắc phục tình trạng overfitting, mà còn giúp cải thiện độ chính xác. BERT đa phần cấu tạo bởi các Neural Network có tính phi tuyến với lượng tham số khổng lồ($BERT_{LARGE}$ có số lượng tham số lên đến 340 triệu), khiến BERT dễ mắc phải tình trạng overfitting. Ensemble là một trong các giải pháp khắc phục tình trạng này. [8] kết hợp các phiên bản BERT-MLP, BERT-CNN và BERT-LSTM trong tác vụ phân loại bài đăng có yếu tố xúc phạm/ganh ghét, kết quả của việc sử dụng Ensemble vượt trội đáng kể. [9] cũng đạt được kết quả ấn tượng khi kết hợp các mô hình BERT với nhau. Điều này cho thấy áp dụng phương pháp Ensemble BERT là một hướng đi hứa hẹn để cải thiện độ chính xác.

III. Mô hình

A. BERT, RoBERTa và PhoBERT

Các nhà nghiên cứu của BERT cho rằng, trong một số tác vụ, thông tin ngữ cảnh được trích xuất từ cả hai hướng là vô cùng quan trọng, họ cải tiến bằng cách cho mô hình này học đồng thời thông tin ngữ cảnh từ cả hai chiều, về cơ bản BERT sử dụng phần Encoder của Transformer. BERT được huấn luyện dựa trên 2 tác vụ học không giám sát: **Masked Language Model** để dự đoán token dựa vào thông tin ngữ

cảnh và **Next Sentence Prediction** để học mối quan hệ giữa hai câu kế tiếp.

RoBERTa [10] là cải tiến dựa trên BERT, các nhà nghiên cứu cho rằng, BERT vẫn chưa được huấn luyện triệt để, họ cải tiến mô hình trên 2 tác vụ học không giám sát, học với bộ dữ liệu lớn hơn và huấn luyện trên batch size lớn hơn. Kết quả cho thấy RoBERTa cải thiện các tác vụ sau tốt hơn.

PhoBERT [6] là RoBERTa phiên bản Tiếng Việt, được huấn luyện trên 20GB dữ liệu và kết quả đạt được khi tinh chỉnh trên các tác vụ sau rất tốt.

Đồ án sử dụng cả 2 phiên bản $PhoBERT_{BASE}$ và $PhoBERT_{LARGE}$ để tinh chỉnh với 2 cấu hình:

- $PhoBERT_{BASE/LARGE} + FeedForward$: sử dụng trạng thái ẩn cuối của token <CLS> kết hợp với MLP.
- $PhoBERT_{BASE/LARGE} + BiLSTM$: sử dụng trạng thái ẩn cuối của token <CLS> kết hợp với BiLSTM.

B. FastText

FastText [11] được sử dụng để trích xuất biểu diễn từ trước khi dữ liệu được đưa vào để huấn luyện mô hình. FastText được huấn luyện dựa trên mô hình skip-gram trong Word2Vec [12]. Gọi $T = \{w_{i-l}, \ldots, w_{i+l}\}$ là một câu trong bộ ngữ liệu, l cho biết độ dài của "cửa sổ trượt", một cửa sổ trượt qua toàn bộ câu có kích thước l. Mỗi từ w_i sẽ có biểu diễn vector là v_i, sẽ là input cho một Neural Network được huấn luyện để dự đoán v_i dựa vào vector ngữ cảnh $C_i = \{v_{i-l}, \ldots, v_{i-1}, v_{i+1}, \ldots, v_{i+l}\}$. Mục đích là để tạo Word Embedding, kết quả là biểu diễn vector của các từ có cùng ý nghĩa sẽ có độ tương đồng cao.

FastText cải tiến hơn Word2Vec khi sử dụng từ phụ (sub-word) khi tạo bộ từ điển. Biểu diễn vector của một từ là sự kết hợp embedding của nhiều từ phụ, điều này giúp khắc phục vấn đề "out of vocabulary" khi từ không tồn tại trong từ điển.

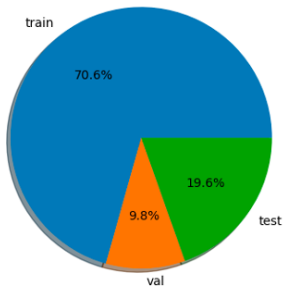
Đồ án sẽ huấn luyện FastText từ đầu, với biểu diễn vector của mỗi từ là một vector 300 chiều. FastText sẽ được kết hợp cùng với 2 cấu hình:

- FastText+BiLSTM: Kết hợp FastText với BiLSTM cũng với số chiều của vector là 300.
- FastText+SVM: Kết hợp FastText với Support Vector Machine, áp dụng PCA để giảm chiều dữ liệu.

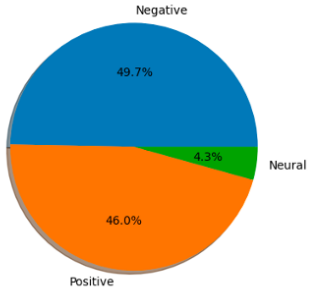
IV. Dữ liệu

Bộ dữ liệu UIT-VSFC [13] (UIT-Vietnamese Student's Feedback Corpus) được sử dụng để huấn luyện và đánh giá mô hình. Bao gồm hơn 16000 câu được gán nhãn tay cho 2 tác vụ: sentiment và topic. Nội dung của bộ dữ liệu chủ yếu là feedback của sinh viên UIT với nhà trường về môn học, bài giảng,... được thu thập mỗi học kì thông qua khảo sát cuối kì từ năm 2013 – 2016. Trong phạm vi đồ án chỉ giải quyết tác vụ sentiment: từ một câu feedback, mô hình sẽ phân loại thuộc nhóm Tích cực (Positive), Tiêu cực (Negative) hay Trung tính (Neural).

Hình 1 cho thấy Dữ liệu chia sẵn thành 3 tập tương ứng với 3 quá trình Train–Validation–Test. Đối với tác vụ sentiment, biểu đồ tròn Hình 2 thống kê số lượng điểm dữ liệu thuộc 3 loại label cho thấy dữ liệu bị mất cân bằng, đa số dữ liệu thuộc nhóm Positive và Negative, điều này có thể ảnh hưởng đến độ chính xác của mô hình khi dự đoán nhóm Neural.

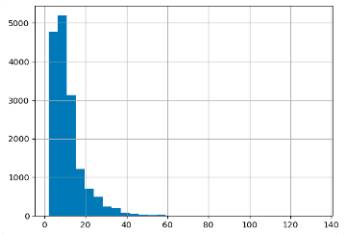


Hình 1: Train - Validation - Test đã được chia sẵn



Hình 2: Phân bố dữ liệu ở 3 label

Vì dữ liệu được lấy từ môi trường giáo dục nên nhìn chung rất sạch sẽ, không cần xử lí thêm nhiều. Sử dụng pyvi để phân đoạn từ và thống kê chiều dài các câu ở Hình 3 cho thấy đa số các câu có chiều dài khoảng 20 token. Chỉ loại bỏ các số/chữ số trong bộ ngữ liệu vì nhóm cho rằng chúng không cần thiết trong tác vụ này.



Hình 3: Chiều dài các câu trong bộ ngữ liệu

V. Thực nghiệm

A. Huấn luyện PhoBERT

Với tác vụ sentiment analysis, PhoBERT có thể được huấn luyện với 2 cách tiếp cận:

- Trích xuất đặc trưng (Feature Extraction): với phương pháp này, PhoBERT đóng vai trò là bộ trích xuất đặc trưng, các tham số được cố định, và output là vector đặc trưng làm input cho lớp Feed Forward/LSTM.

- Tinh chỉnh (Fine tuning): trong phương pháp này, các tham số của PhoBERT được tinh chỉnh đồng thời với các lớp kế tiếp trong quá trình huấn luyện.

Thực nghiệm cho thấy cách tinh chỉnh hiệu quả hơn, sử dụng Cross Entropy Loss là hàm loss, thuật toán tối ưu sử dụng Adam [14] với learning rate là 1×10^{-4}. LogSoftmax làm activation function cuối bởi thực nghiệm cho thấy LogSoftmax giúp loss giảm tốt hơn. Huấn luyện trên tập Train, thêm cơ chế Early Stopping trên tập Validation để hạn chế overfitting, áp dụng Gradient Clipping tránh Gradient Exploding. Ngoài ra, khi huấn luyện phiên bản $PhoBERT_{LARGE}$ trên Google Colab luôn bị tràn RAM do lượng tham số quá lớn, áp dụng Gradient Accumulation để train mô hình được batch size lớn hơn. Cài đặt OneCycle [15] learning rate scheduling để thay đổi learning rate trong khi huấn luyện.

B. Huấn luyện FastText

FastText đóng vai trò trích xuất word embedding trước khi đưa vào bộ phân loại. Huấn luyện FastText trên bộ Train và Validation với số chiều vector là 300.

Đối với mô hình FastText-BiLSTM, word embedding sẽ được đưa vào BiLSTM , sau đó được feed vào LogSoftmax. Cũng áp dụng Adam với learning rate là 1×10^{-3}, OneCycleLR, Gradient Clipping, Early Stopping để đảm bảo mô hình không bị overfitting, vanishing hay exploding gradient.

Mô hình FastText-SVM quy trình phức tạp hơn, SVM được thiết kế để xử lí dữ liệu dạng bảng, đòi hỏi số feature của các sample phải bằng nhau. Vì vậy giới hạn chiều dài mỗi câu là 18 token, sau đó trích xuất embedding thu được vector có $18 \times 300 = 5400$ chiều. Để giảm thời gian huấn luyện SVM, áp dụng PCA để giảm chiều dữ liệu xuống còn 800 chiều, sau đó đưa vào SVM để phân loại. Theo [16] quan sát, sau khi áp dụng PCA, 7 vector riêng đầu tiên của word embedding có độ tương đồng cao cho tất cả các word, họ đề xuất loại bỏ các vector này để tăng tính phân biệt cho các embedding, thực nghiệm cho thấy kết quả không được cải thiện.

C. Phương pháp Ensemble

Phương pháp Ensemble đề cập đến tập hợp nhiều mô hình cùng hoạt động để đưa ra dự đoán cuối cùng. Thường được sử dụng để khắc phục vấn đề Overfitting trên Decision Tree, giúp cải thiện độ chính xác. Bagging là một trong các kĩ thuật Ensemble, dự đoán là kết quả trung bình cộng của nhiều mô hình. Bagging kết hợp 2 mô hình theo công thức:

$$ensemble_{pred} = ratio \times pred_1 + (1 - ratio) \times pred_2$$

Trong đó:

- $pred_1$ và $pred_2$ lần lượt là output của 2 mô hình khác nhau trước khi feed vào LogSoftmax.
- $ratio$ cho biết mức độ đóng góp của mô hình vào kết quả cuối, giá trị trong khoảng [0,1].

VI. KẾT QUẢ

A. Kết quả trên tập Test

Bảng I cho thấy $PhoBERT_{BASE}$ đem lại các kết quả tốt nhất trên cả 2 cấu hình FeedForward(1) và LSTM(3).

Model	Precision	Recall	F1
(1) PhoBERT (base)+FeedForward	0.92502	0.92988	0.92348
(2) PhoBERT (large)+FeedForward	0.91447	0.90935	0.88475
(3) PhoBERT (base)+BiLSTM	0.92399	0.92893	0.92259
(4) PhoBERT (large)+BiLSTM	0.91062	0.90556	0.88104
(5) FastText+BiLSTM	0.84022	0.86323	0.84127
(6) FastText+SVM	0.84825	0.86639	0.85023

Bảng I: Kết quả của các mô hình trên tập Test

$PhoBERT_{LARGE}$ dù có số lượng tham số khổng lồ nhưng kết quả thấp hơn đáng kể so với phiên bản BASE, nguyên nhân có thể do mô hình bị overfitting. SVM(6) có kết quả ấn tượng khi cùng sử dụng FastText nhưng có kết quả cao hơn BiLSTM. Điều đó cho thấy mô hình Machine Learning truyền thống vẫn đủ tốt ở thời điểm hiện tại.

B. Kết quả Ensemble trên tập Test

Model	Ratio	Precision	Recall	F1
(2) + (6)	0.5	0.89417	0.91124	0.88877
(2) + (4)	0.7	0.91587	0.91093	0.88627
(2) + (5)	0.8	0.91521	0.91030	0.88565
(4) + (6)	0.2	0.89082	0.90556	0.88562
(4) + (5)	0.7	0.91145	0.90651	0.88195
(5) + (6)	0.4	0.85532	0.87208	0.85340

Bảng II: Kết quả Ensemble các mô hình trên tập Test

Bảng II cho thấy có vẻ Ensemble không giúp cải thiện hiệu năng của mô hình, thậm chí giảm, không có sự kết hợp nào cải tiến.

Kết quả phân tích trên Confusion Matrix cho thấy nguyên nhân do hiện tượng mất cân bằng dữ liệu. Đa số các dự đoán đúng trên nhóm Positive và Negative, các dự đoán đúng trên Neural rất thấp, hầu như bằng 0.

Giải pháp là thêm loss weight vào Cross Entropy Loss, nhãn có tần suất xuất hiện thấp sẽ thêm trọng số lớn, để loss của nhãn đó lớn hơn, optimizer sẽ phạt gradient nặng hơn.

C. Kết quả trên tập Test sau khi thêm loss weight

Model	Precision	Recall	F1
(1) PhoBERT (base)+FeedForward	0.92867	0.92672	0.92751
(2) PhoBERT (large)+FeedForward	0.90756	0.9024	0.87796
(3) PhoBERT (base)+LSTM	0.92489	0.92356	0.92407
(4) PhoBERT (large)+LSTM	0.90965	0.90461	0.8801
(5) FastText+LSTM	0.85727	0.81207	0.83015
(6) FastText+SVM	0.85376	0.86229	0.85561

Bảng III: Kết quả của các mô hình trên tập Test sau khi thêm loss weight

Thêm loss weight thực sự giúp cải thiện độ chính xác, Bảng III cho thấy hiệu năng các mô hình $PhoBERT_{BASE}$ và SVM tăng, tốt nhất cải thiện lên đến 0.4%, tuy hiệu năng các mô hình sử dụng $PhoBERT_{LARGE}$ bị giảm.

D. Kết quả Ensemble trên tập Test sau khi thêm loss weight

Kết quả áp dụng Ensemble từ Bảng IV có hiệu quả, hầu hết các sự kết hợp đều giúp cải thiện độ chính xác. Kết quả tốt nhất tăng thêm 0.1% so với trước. Đáng chú ý nhất là sự kết hợp

Model	Ratio	Precision	Recall	F1
(1) + (4)	0.8	0.92845	0.92956	0.92889
(1) + (2)	0.9	0.92899	0.92798	0.92837
(1) + (6)	0.5	0.92932	0.92830	0.92830
(1) + (5)	0.9	0.92943	0.92672	0.92783
(3) + (4)	0.8	0.92507	0.92704	0.92584
(3) + (6)	0.8	0.92545	0.92451	0.92484

Bảng IV: Kết quả Ensemble các mô hình trên tập Test sau khi thêm loss weight

của (1)$PhoBERT_{BASE}$+FeedForward với (6)FastText+SVM. Điều này cho thấy các thuật toán Machine Learning ở thời điểm hiện tại không những vẫn còn hiệu quả, mà kết hợp với chúng còn giúp tăng độ chính xác.

VII. Kết luận

Trong đồ án này nhóm đã hiệu chỉnh được PhoBERT trên bộ dữ liệu Tiếng Việt và cho thấy kết quả ấn tượng hơn cách tiếp cận sử dụng FastText. Phương pháp Ensemble tỏ rất hiệu quả khi giúp tăng độ chính xác trên tập Test.

Thuật toán SVM vẫn rất tốt, không những có kết quả tốt hơn LSTM khi cùng sử dụng FastText, mà kết hợp với PhoBERT giúp tăng độ chính xác.

Tài liệu

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[3] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 2227–2237. [Online]. Available: https://aclanthology.org/N18-1202

[4] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.

[5] Q. T. Nguyen, T. L. Nguyen, N. H. Luong, and Q. H. Ngo, "Fine-tuning bert for sentiment analysis of vietnamese reviews," in *2020 7th NAFOSTED Conference on Information and Computer Science (NICS)*. IEEE, 2020, pp. 302–307.

[6] D. Q. Nguyen and A. T. Nguyen, "Phobert: Pre-trained language models for vietnamese," *arXiv preprint arXiv:2003.00744*, 2020.

[7] K. Quoc Tran, A. Trong Nguyen, P. G. Hoang, C. D. Luu, T.-H. Do, and K. Van Nguyen, "Vietnamese hate and offensive detection using phobert-cnn and social media streaming data," *Neural Computing and Applications*, pp. 1–22, 2022.

[8] K. Mnassri, P. Rajapaksha, R. Farahbakhsh, and N. Crespi, "Bert-based ensemble approaches for hate speech detection," *arXiv preprint arXiv:2209.06505*, 2022.

[9] H. Dang, K. Lee, S. Henry, and O. Uzuner, "Ensemble bert for classifying medication-mentioning tweets," in *Proceedings of the Fifth Social Media Mining for Health Applications Workshop & Shared Task*, 2020, pp. 37–41.

[10] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[11] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the association for computational linguistics*, vol. 5, pp. 135–146, 2017.

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[13] K. Van Nguyen, V. D. Nguyen, P. X. Nguyen, T. T. Truong, and N. L.-T. Nguyen, "Uit-vsfc: Vietnamese students' feedback corpus for sentiment analysis," in *2018 10th international conference on knowledge and systems engineering (KSE)*. IEEE, 2018, pp. 19–24.

[14] P. Kingma Diederik and J. B. Adam, "A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[15] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," in *Artificial intelligence and machine learning for multi-domain operations applications*, vol. 11006. SPIE, 2019, pp. 369–386.

[16] V. Raunak, V. Gupta, and F. Metze, "Effective dimensionality reduction for word embeddings," in *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, 2019, pp. 235–243.