Sentiment Analysis

Huỳnh Chấn Huy 19521609-UIT
Nguyễn Đặng Nhật Hào 20520490-UIT
Bùi Nguyễn Anh Trung 20520332-UIT

1 Giới thiệu

Việc thể hiện cảm xúc là một nhu cầu cơ bản của con người. Ngoài việc biểu hiện qua giọng nói và gương mặt, việc lựa chọn từ ngữ phù hợp cũng đóng vai trò quan trọng trong việc truyền tải cảm xúc một cách rõ ràng và chính xác. Đặc biệt gần đây các mạng xã hội đã trở nên phổ biến và con người càng có nhu cầu thể hiện cảm xúc qua những đoạn văn bản. Từ đó, việc nhận diện cảm xúc qua văn bản sẽ đem lại nhiều lợi ích như: báo cáo trải nghiệm người dùng, đánh giá sản phẩm và dịch vụ, ưu tiên xử lí những vấn đề khẩn cấp.....
Trong bài này sẽ sử dụng bộ dữ liệu UIT-VSMEC để nhận dạng cảm xúc qua các bình luận tiếng Việt trên mạng xã hội. Mục tiêu nghiên cứu là tìm các giải pháp có kết quả tốt hơn kết quả được công bố.

2 Dữ liệu

Hiện nay, các bộ dữ liệu về Sentiment Analysis trong tiếng Việt là khá hiếm. Trong đó, UIT-VSMEC nổi bật với việc cung cấp 7 nhãn cảm xúc: enjoyment, sadness, anger, surprise, fear, disgust và other, đem lại nhiều thách thức hơn so với các bộ dữ liệu chỉ có 3 nhãn cảm xúc: tích cực, tiêu cực và trung tính. Điều này mang lại sự đa dạng hơn trong việc nghiên cứu và phát triển các mô hình Sentiment Analysis trong tiếng Việt.
UIT-VSMEC là một bộ dữ liệu được thu thập và xây dựng bởi nhóm nghiên cứu NLP (Xử lý Ngôn ngữ Tự nhiên) thuộc Trường Đại học Công nghệ Thông tin - Đại học Quốc gia Thành phố Hồ Chí Minh.

No.	Vietnamese sentences	Emotion
1	Ảnh đẹp quá!	Enjoyment
2	Tao khóc..huhu.. Tao rớt rồi	Sadness
3	ở dơ vãi - _ -	Disgust

Table 1: Các bình luận được gán nhãn cảm xúc

UIT-VSMEC bao gồm 6.927 câu đã được gán nhãn cảm xúc với một trong bảy nhãn cảm xúc. Quá trình gán nhãn được thực hiện thủ công đảm bảo tính chính xác và đáng tin cậy của dữ liệu. Đây là một bộ dữ liệu quan trọng giúp nghiên cứu và phát triển các mô hình Sentiment Analysis cho tiếng Việt.

Emotion	Sentences	Percentage (%)
Enjoyment	1,965	28.36
Disgust	1,338	19.31
Sadness	1,149	16.59
Anger	480	6.92
Fear	395	5.70
Surprise	309	4.46
Other	1,291	18.66
Total	**6,927**	**100**

Table 2: Thống kê các nhãn cảm xúc

Dữ liệu UIT-VSMEC đặt ra một số thách thức trong việc phân loại cảm xúc:
- Mặc dù UIT-VSMEC là một bộ dữ liệu đáng kể, nhưng có thể chưa đủ lớn để đại diện hết cho tất cả các nhãn cảm xúc.
- Dữ liệu UIT-VSMEC chứa xen lẫn nhiều loại ngôn ngữ, bao gồm cả từ viết tắt, teencode và sai chính tả. Điều này đòi hỏi phải phải xử lí dữ liệu và định nghĩa ý nghĩa của các từ và cụm từ không tiêu chuẩn trước khi đưa vào mô hình.
- 7 nhãn cảm xúc khác nhau, đôi khi con người cũng gặp khó khăn trong việc phân biệt rõ ràng giữa các loại cảm xúc. Vì vậy, việc phân loại đúng và chính xác các câu có cảm xúc khác nhau là một thách thức đối với mô hình Sentiment Analysis.

3 Phương pháp

3.1 Các kỹ thuật máy học

Sử dụng các kỹ thuật vector hóa văn bản như Bag of Words, TFIDF hoặc N-grams để biểu diễn văn

bản dưới dạng vector số. Sau đó đưa vào các mô hình máy học đơn giản như Logistic Regression, NaiveBayes, Glove vectors để học từ các vector và dự đoán cảm xúc. Bên cạnh đó thử loại bỏ các từ stopword để xem độ hiệu quả Phần lớn các thuật toán học máy không hiểu dữ liệu văn bản mà là dữ liệu số. Word embedding là một cách tiếp cận để biểu diễn các từ ở dạng vector. Nó cung cấp các biểu diễn vector tương tự cho các từ có nghĩa tương tự. Nó giúp mô hình nắm bắt được ý nghĩa ngôn ngữ của từ. Các phương pháp phổ biến: Word2Vec, GloVe, FastText, ELMo.

3.1.1 Word2vec

Word2Vec và GloVe cung cấp các biểu diễn vector riêng biệt cho các từ trong từ vựng. Điều này dẫn đến sự thiếu hiểu biết về cấu trúc bên trong của ngôn ngữ. Đây là một hạn chế đối với ngôn ngữ giàu hình thái vì nó bỏ qua mối quan hệ cú pháp của các từ. Mô hình sẽ không hiểu được từ mới nếu không có trong từ điển từ trước. Có 2 mô hình tính toán: Skip-gram và Túi từ (CBOW).

Mô hình Skip-gram Mô hình skip-gam giả định rằng một từ có thể được sử dụng để sinh ra các từ xung quanh nó trong một chuỗi văn bản. Mô hình Skip-gram, với từ trung tâm cho trước, các từ ngữ cảnh được sinh ra độc lập với nhau. VD: với câu 'I want to learn FastText.' mô hình skip-gram quan tâm đến xác suất có điều kiện sinh ra các từ ngữ cảnh ("I", "want", "to" và "FastText") nằm trong khoảng cách không quá 2 từ: $P(\text{"I"}, \text{"want"}, \text{"to"}, \text{"FastText"} \mid \text{"learn"})$. Ta giả định rằng, với từ đích trung tâm cho trước, các từ ngữ cảnh được sinh ra độc lập với nhau.
Trong mô hình skip-gam, mỗi từ được biểu diễn bằng hai vector d chiều để tính xác suất có điều kiện. Giả sử chỉ số của một từ trong từ điển là i, vector của từ được biểu diễn là $\mathbf{v}_i \in \mathbb{R}^d$ khi từ này là từ đích trung tâm và là $\mathbf{u}_i \in \mathbb{R}^d$ khi từ này là một từ ngữ cảnh. Gọi c và o lần lượt là chỉ số của từ đích trung tâm w_c và từ ngữ cảnh w_o trong từ điển. Có thể thu được xác suất có điều kiện sinh ra từ ngữ cảnh cho một từ đích trung tâm cho trước bằng phép toán softmax trên tích vô hướng của vector:

$$P(w_o \mid w_c) = \frac{\exp(\mathbf{u}_o^\top \mathbf{v}_c)}{\sum_{i \in \mathcal{V}} \exp(\mathbf{u}_i^\top \mathbf{v}_c)}$$

trong đó, tập chỉ số trong bộ từ vựng là $\mathcal{V} = \{0, 1, \ldots, |\mathcal{V}| - 1\}$. Giả sử trong một chuỗi văn bản có độ dài T, từ tại bước thời gian t được ký hiệu là $w^(t)$. Giả sử rằng các từ ngữ cảnh được sinh độc lập với từ trung tâm cho trước. Khi kích thước cửa sổ ngữ cảnh là m, hàm hợp lý (likelihood) của mô hình skip-gam là xác suất kết hợp sinh ra tất cả các từ ngữ cảnh với bất kỳ từ trung tâm cho trước nào

$$\prod_{t=1}^{T} \prod_{-m \le j \le m,\ j \ne 0} P(w^{(t+j)} \mid w^{(t)})$$

Ở đây, bất kỳ bước thời gian nào nhỏ hơn 1 hoặc lớn hơn T đều có thể được bỏ qua.

Mô hình Túi từ Liên tục (CBOW) Mô hình túi từ liên tục (Continuous bag of words - CBOW) tương tự như mô hình skip-gram. Khác biệt lớn nhất là mô hình CBOW giả định rằng từ đích trung tâm được tạo ra dựa trên các từ ngữ cảnh phía trước và sau nó trong một chuỗi văn bản. Với cùng một chuỗi văn bản gồm các từ "I", "want", "to", "learn", và "FastText", trong đó "learn" là từ đích trung tâm, với kích thước cửa sổ ngữ cảnh bằng 2, mô hình CBOW quan tâm đến xác suất có điều kiện để sinh ra từ đích "learn" dựa trên các từ ngữ cảnh "I", "want", "to" và "FastText" như sau: $P(\text{"learn"} \mid \text{"I"}, \text{"want"}, \text{"to"}, \text{"FastText"})$
Vì có quá nhiều từ ngữ cảnh trong mô hình CBOW, ta sẽ lấy trung bình các vector từ của chúng và sau đó sử dụng phương pháp tương tự như trong mô hình skip-gram để tính xác suất có điều kiện. Giả sử $\mathbf{v_i} \in \mathbb{R}^d$ và $\mathbf{u_i} \in \mathbb{R}^d$ là vector từ ngữ cảnh và vector từ đích trung tâm của từ có chỉ số i trong từ điển (lưu ý rằng các ký hiệu này ngược với các ký hiệu trong mô hình skip-gram). Gọi c là chỉ số của từ đích trung tâm w_c, và $o_1, \ldots, o_{2m}$ là chỉ số các từ ngữ cảnh $w_{o_1}, \ldots, w_{o_{2m}}$ trong từ điển. Do đó, xác suất có điều kiện sinh ra từ đích trung tâm dựa vào các từ ngữ cảnh cho trước là:

$$P(w_c \mid w_{o_1}, \ldots, w_{o_{2m}}) =$$

$$\frac{\exp\left(\frac{1}{2m}\mathbf{u}_c^\top(\mathbf{v}_{o_1} + \ldots, +\mathbf{v}_{o_{2m}})\right)}{\sum_{i \in \mathcal{V}} \exp\left(\frac{1}{2m}\mathbf{u}_i^\top(\mathbf{v}_{o_1} + \ldots, +\mathbf{v}_{o_{2m}})\right)}$$

Cho một chuỗi văn bản có độ dài T, ta giả định rằng từ xuất hiện tại bước thời gian t là $w^(t)$, và kích thước của cửa sổ ngữ cảnh là m. Hàm hợp lý của mô hình CBOW là xác suất sinh ra bất kỳ từ đích trung tâm nào dựa vào những từ ngữ cảnh.

$$\prod_{t=1}^{T} P(w^{(t)} \mid w^{(t-m)}, \ldots, w^{(t-1)}, w^{(t+1)}, \ldots, w^{(t+m)})$$

3.1.2 FastText

Trong word2vec, ta không trực tiếp sử dụng thông tin hình thái học. Trong cả mô hình skip-gram và túi từ (bag-of-word) liên tục, ta sử dụng các vector khác nhau để biểu diễn các từ ở các dạng khác nhau. Chẳng hạn, "dog" và "dogs" được biểu diễn bởi hai vector khác nhau, trong khi mối quan hệ giữa hai vector đó không biểu thị trực tiếp trong mô hình. Từ quan điểm này, fastText [Bojanowski et al., 2017] đề xuất phương thức embedding từ con (subword embedding), thông qua việc thực hiện đưa thông tin hình thái học vào trong mô hình skip-gram trong word2vec.

Trong fastText, mỗi từ trung tâm được biểu diễn như một tập hợp của các từ con. Dưới đây ta sử dụng từ "where" làm ví dụ để hiểu cách các từ tố được tạo thành. Trước hết, ta thêm một số ký tự đặc biệt "<" và ">" vào phần bắt đầu và kết thúc của từ để phân biệt các từ con được dùng làm tiền tố và hậu tố. Rồi ta sẽ xem từ này như một chuỗi các ký tự để trích xuất n-grams. Chẳng hạn, khi n=3, ta có thể nhận tất cả từ tố với chiều dài là 3 "<wh", "whe", "her", "ere", "re>", và từ con đặc biệt "<where>". Trong fastText, với một từ w, ta ghi tập hợp của tất cả các từ con của nó với chiều dài từ 3 đến 6 và các từ con đặc biệt là $\mathcal{G}_w$. Do đó, từ điển này là tập hợp các từ con của tất cả các từ. Giả sử vector của từ con g trong từ điển này là z_g. Thì vector từ trung tâm u_w cho từ w trong mô hình skip-gram có thể biểu diễn là

$$\mathbf{u}_w = \sum_{g \in \mathcal{G}_w} \mathbf{z}_g$$

Phần còn lại của tiến trình xử lý trong fastText giống với mô hình skip-gram. So sánh với mô hình skip-gram, từ điển của fastText lớn hơn dẫn tới nhiều tham số mô hình hơn. Hơn nữa, vector của một từ đòi hỏi tính tổng của tất cả vector từ con dẫn tới độ phức tạp tính toán cao hơn. Tuy nhiên, ta có thể thu được các vector tốt hơn cho nhiều từ phức hợp ít thông dụng, thậm chí cho cả các từ không hiện diện trong từ điển này nhờ tham chiếu tới các từ khác có cấu trúc tương tự.

3.2 Các kỹ thuật học sâu

3.2.1 BERT

TRAMSFORMER

- Vào năm 2017, Transformer ra đời và là một trong những nghiên cứu đột phá của lĩnh vực NLP vào thời điểm đó.
- 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-termdependency 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.

Kiến trúc

- Kiến trúc của mô hình BERT là một kiến trúc đa tầng gồm nhiều lớp Bidirectional Transformer encoder
- BERT BASE : L=12, H=768, A=12, Total Parameters=110M
- BERT LARGE : L=24, H=1024, A=16, Total Parameters=340M

L : số lớp Transformer (blocks) H : kích thước của các lớp ẩn A : số head ở lớp Attention Kích thước bộ lọc (filter-size) : 4H

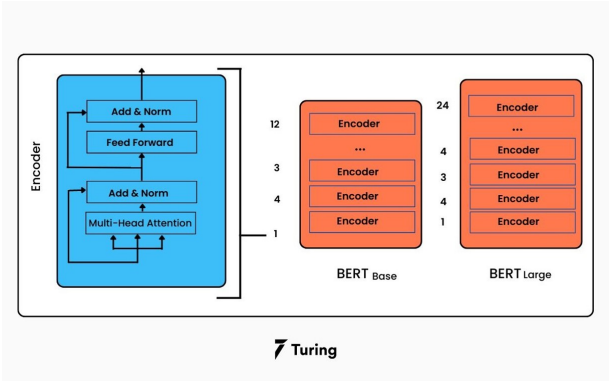

Figure 1: Minh họa các kiến trúc BERT

BERTBASE đã được chọn để có một kích thước mô hình giống hệt như mô hình OpenAI GPT để nhằm mục đích so sánh giữa 2 mô hình này. Tuy nhiên, một cách đơn giản để so sánh, BERT Transformer sử dụng các attention 2 chiều trong khi GPT Transformer sử dụng các attention 1 chiều(không tự nhiên, không hợp với cách mà xuất hiện của ngôn ngữ), nơi mà tất cả các từ chỉ chú ý tới ngữ cảnh trái của nó.

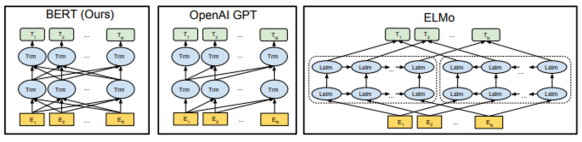

Figure 2: So ánh kiến trúc BERT, GPT và ELMO

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-toright 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.
Chúng ta đào tạo BERT bằng cách sử dụng 2 nhiệm vụ dự đoán không giám sát được gọi là Masked LM và Next Sentence Prediction. Cả 2 sẽ được trình bày ngay trong phần nội dung dưới đây.

Biểu diễn đầu vào Có thể là biểu diễn của một câu văn bản đơn hoặc một cặp câu văn bản(ví dụ: [Câu hỏi, câu trả lời]) được đặt thành 1 chuỗi tạo bởi các từ.

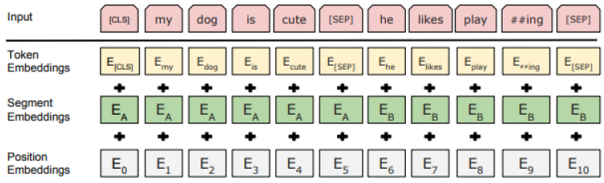

Figure 3: Minh họa các biểu diễn đầu vào của BERT

Khi có một chuỗi đầu vào cụ thể, biểu diễn đầu vào của chúng ta được xây dựng bằng cách tính tổng các token đó với vector phân đoạn và vị trí tương ứng của các từ trong chuỗi. Cho dễ hình dung, biểu diễn đầu vào được trực quan hóa trong hình dưới đây: Chúng ta sử dụng WordPiece embeddings (Wu et al., 2016) với một từ điển 30.000 từ và sử dụng ## làm dấu phân tách. Ví dụ từ playing được tách thành play##ing. Chúng ta sử dụng positional embeddings với độ dài câu tối đa là 512 tokens. Token đầu tiên cho mỗi chuỗi được mặc định là một token đặc biệt có giá trị là [CLS]. Đầu ra của Transformer(hidden state cuối cùng) tương ứng với token này sẽ được sử dụng để đại diện cho cả câu trong các nhiệm vụ phân loại. Nếu không trong các nhiệm vụ phân loại, vector này được bỏ qua. Trong trường hợp các cặp câu được gộp lại với nhau thành một chuỗi duy nhất, chúng ta phân biệt các câu theo 2 cách. Đầu tiên, chúng ta tách chúng bởi một token đặc biệt [SEP]. Thứ hai, chúng ta thêm một segment embedding cho câu A và một segment embedding khác cho câu B như hình vẽ. Khi chỉ có 1 câu đơn duy nhất, segment embedding của chúng ta chỉ có cho câu A.

Mô hình Ngôn ngữ có mặt nạ (MarkedLM) Một mô hình ngôn ngữ dự đoán một token bằng cách sử dụng ngữ cảnh phía bên trái của nó. Để mã hóa ngữ cảnh hai chiều khi biểu diễn mỗi token, BERT ngẫu nhiên che mặt nạ các token và sử dụng các token lấy từ ngữ cảnh hai chiều để dự đoán các token mặt nạ đó. Tác vụ này được gọi là mô hình hóa ngôn ngữ có mặt nạ.
Trong tác vụ tiền huấn luyện này, 15% số token sẽ được lựa chọn ngẫu nhiên để làm các token mặt nạ cho việc dự đoán. Để dự đoán một token mặt nạ mà không sử dụng nhãn, một hướng tiếp cận đơn giản là luôn luôn thay thế nó bằng token đặc biệt "<mask>" trong chuỗi đầu vào BERT. Tuy nhiên, token "<mask>" sẽ không bao giờ xuất hiện khi tinh chỉnh. Để tránh sự không đồng nhất giữa tiền huấn luyện và tinh chỉnh, nếu một token được che mặt nạ để dự đoán (ví dụ, từ "great" được chọn để che mặt nạ và dự đoán trong câu "this movie is great"), trong đầu vào nó sẽ được thay thế bởi:

- token đặc biệt "<mask>", 80% số lần (ví dụ, "this movie is great" trở thành "this movie is <mask>")
- token ngẫu nhiên, 10% số lần (ví dụ, "this movie is great" trở thành "this movie is drink")
- chính token đó, 10% số lần (ví dụ, "this movie is great" trở thành "this movie is great")

Lưu ý rằng trong 15% token được chọn để che mặt nạ, 10% số token đó sẽ được thay thế bằng một token ngẫu nhiên. Việc thi thoảng thêm nhiễu sẽ giúp BERT giảm thiên kiến về phía token có mặt nạ (đặc biệt khi token nhãn không đổi) khi mã hóa ngữ cảnh hai chiều.

Dự đoán câu tiếp theo (Next Sentence Prediction) Mặc dù mô hình hóa ngôn ngữ có mặt nạ có thể mã hóa ngữ cảnh hai chiều để biểu diễn từ ngữ, nó không thể mô hình hóa các mối quan hệ logic giữa các cặp văn bản một cách tường minh. Để hiểu hơn về mối quan hệ giữa hai chuỗi văn bản, BERT sử dụng tác vụ phân loại nhị phân, dự đoán câu tiếp theo (next sentence prediction) trong quá trình tiền huấn luyện. Khi sinh các cặp câu cho quá trình tiền huấn luyện, một nửa trong số đó là các cặp câu liên tiếp nhau trong thực tế và được gán nhãn "Đúng" (True); và trong nửa còn lại, câu thứ hai được lấy mẫu ngẫu nhiên từ kho ngữ liệu và cặp này được gán nhãn "Sai" (False).

Fine-tuning Procedure Phân loại câu, BERT được fine-tuning rất đơn giản. Để có được biểu diễn của một chuỗi đầu vào với số chiều cố định, chúng ta chỉ cần lấy hidden state ở lớp cuối cùng,

tức là đầu ra của lớp Transformer cho token đầu tiên (token đặc biệt [CLS] được xây dựng cho đầu chuỗi). Chúng ta gọi vector này là $C(C \in R^H)$.

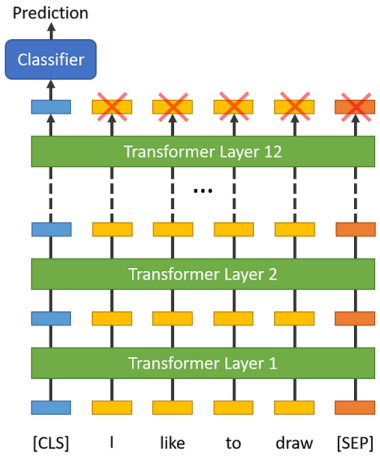

Figure 4: Minh họa sử dụng lớp ngoài cùng của token [CLS] để phân loại

Chỉ có 1 tham số được thêm vào trong quá trình fine-tuning là $W \in R^{K*H}$ với K là số nhãn lớp phân loại. Xác suất của nhãn P là một phân phối với $P \in R^K$ được tính toán bằng một hàm softmax $P = softmax(C * W^T)$. Tất cả các tham số của BERT và W được fine-tuning để tối ưu hóa hàm lỗi.

3.2.2 RoBERTa

RoBERTa được giới thiệu bởi Facebook là một phiên bản được huấn luyện lại của BERT với một phương pháp huấn luyện tốt hơn với dữ liệu được tăng gấp 10 lần. Để tăng cường quá trình huấn luyện, RoBERTa không sử dụng cơ chế dự đoán câu kế tiếp (NSP) từ BERT mà sử dụng kỹ thuật mặt nạ động (dynamic masking), theo đó các token mặt nạ sẽ bị thay đổi trong quá trình huấn luyện. Sử dụng kích thước batch lớn hơn cho thấy hiệu quả tốt hơn khi huấn luyện.

3.2.3 PhoBERT

Được xây dựng dựa trên kiến trúc của BERT để tạo ra các vector đặc trưng của văn bản cho các nhiệm vụ như:

- Tìm từ đồng nghĩa, trái nghĩa, cùng nhóm dựa trên khoảng cách của từ trong không gian biểu diễn đa chiều.
- Xây dựng các vector embedding cho các tác vụ NLP như sentiment analysis, phân loại văn bản.
- Gợi ý từ khóa tìm kiếm trong các hệ thống search.

Có 3 phiên bản:

- vinai/phobert-base với 135M parameters
- vinai/phobert-large với 370M parameters
  Cả 2 đều train với 20GB dữ liệu: 1GB Vietnamese Wikipedia corpus, 19GB Vietnamese news corpus.
- vinai/phobert-base-v2 với 135M parameters.
  Train trên 20GB dữ liệu và 120GB OSCAR-2301.

3.2.4 Biểu diễn ngữ cảnh của Transformer

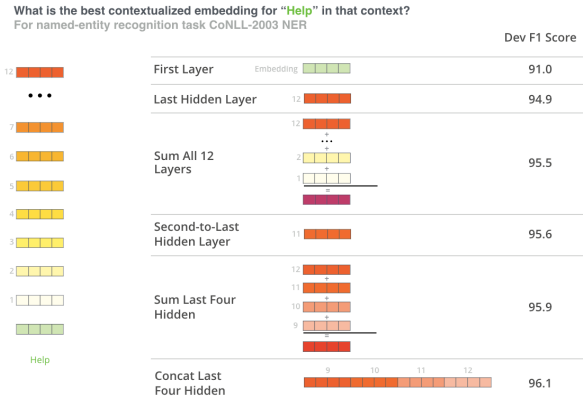

Figure 5: Minh họa các cách tận dụng các lớp đầu ra để tối ưu tác vụ

Chúng ta đã quen với cách tiêu chuẩn của Fine-tuning: chỉ đơn giản là thêm một lớp đầu ra bổ sung sau Transformer cho các tác vụ phía dưới hoặc phần cuối của các mô hình, trong đó lấy các biểu diễn từ lớp cuối cùng của các mô hình ngôn ngữ được huấn luyện trước đó làm đầu vào mặc định.
Tuy nhiên, do cấu trúc đa tầng của Transformers, các lớp khác nhau sẽ bắt được các cấp độ biểu diễn khác nhau. Chúng học được một hệ thống phân cấp giàu thông tin ngôn ngữ, ví dụ như các đặc trưng bề mặt ở các tầng thấp, các đặc trưng cú pháp ở các tầng trung và các đặc trưng ngữ nghĩa ở các tầng cao hơn.
Các tác giả của BERT đã thử nghiệm các chiến lược word embedding bằng cách đưa các kết hợp vector khác nhau làm tính năng đầu vào cho một BiLSTM được sử dụng trong một tác vụ nhận dạng thực thể được đặt tên và quan sát các điểm số F1 kết quả. Sự kết hợp của 4 lớp cuối cùng đã cho ra kết quả tốt nhất.
Điều này được chứng minh một phần bằng việc nhận thấy rằng các lớp khác nhau của BERT mã hóa các loại thông tin rất khác nhau, do đó chiến lược pooling phù hợp sẽ thay đổi tùy thuộc vào

ứng dụng vì các lớp khác nhau mã hóa các loại thông tin khác nhau. Điều này cũng đúng cho các biến thể khác.

CLS Embeddings BERTBASE : L=12, H=768, A=12

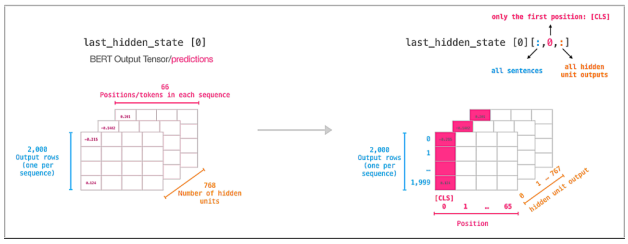

Figure 6: Minh họa sử dụng lớp đầu ra của token [CLS] để phân loại

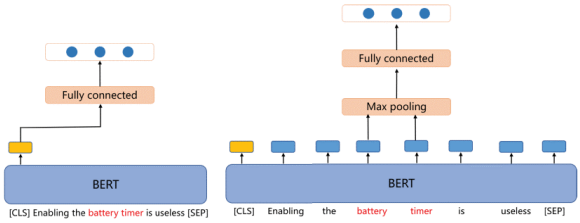

Figure 7: Minh họa sử dụng trung bình và giá trị tối đa của các token ở lớp ngoài cùng để phân loại

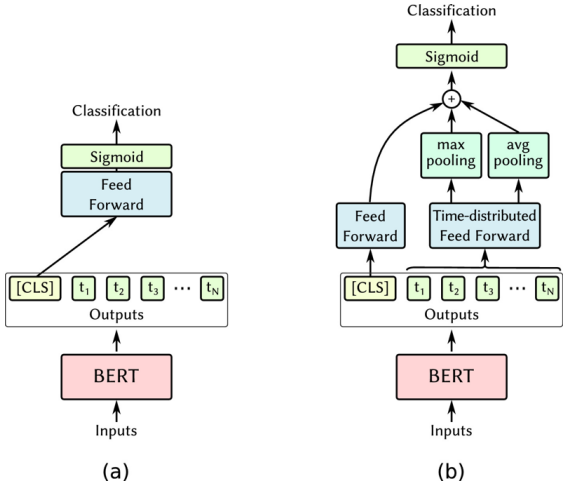

Figure 8: Minh họa kết hợp giá trị trung bình token, giá trị lớn nhất các token và giá trị token [CLS] để phân loại

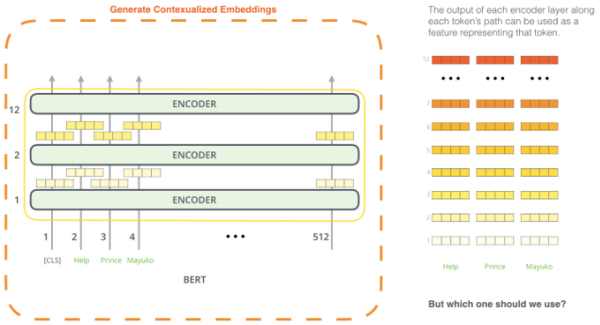

Figure 9: Minh họa sử dụng thông tin các tầng khác để phân loại

4 Kết quả

4.1 Các thuật toán máy học

Model	Accuracy
Logistic + BOW	0.57
Fastext + Naive Bayes	0.516
NaiveBayes + BOW	0.52
Naive Bayes + TFIDF	0.47
Naive Bayes + Ngrams	0.49
Cleaned text Naive Bayes + BOW	0.49

Các mô hình máy học tuy không có độ chính xác cao nhưng khá tốt với các phương pháp kháckhác. Tốt nhất là mô hình logistic.

4.2 Các mô hình học sâu

Model	Accuracy
RoBERTa	0.51
GPT-2	0.5
PhoBert	0.6

Ở hai mô hình RoBERTa và GPT-2 thì không tốt như PhoBERT bởi vì PhoBERT được học trên ngôn ngữ Tiếng Việt.

4.3 Mở rộng các mô hình BERT

Model	Accuracy
PhoBert-base	0.6
PhoBert-mean	0.6
PhoBert-max	0.61
PhoBert-meanmax	0.6
PhoBert-layer11	0.6
PhoBert-concat4lastlayer	0.61

Thay đổi kiến trúc của PhoBERT đã giúp mô hình vượt ngưỡng độ chính xác 0.6

Việc có nhiều mô hình chỉ đạt ngưỡng 0.6 thể hiện kết quả không còn phụ thuộc quá nhiều vào mô hình, đây có thể là ngưỡng giới hạn của bộ dữ liệu.

5 Kết luận

Các mô hình embedding từ như word2vec và GloVe có tính chất độc lập với ngữ cảnh. Hai mô hình này gán cùng một vector được tiền huấn luyện cho cùng một từ bất kể ngữ cảnh xung quanh của từ đó là gì (nếu có). Do đó, rất khó để các mô hình này xử lý tốt các trường hợp phức tạp về ngữ nghĩa hay đa nghĩa trong các ngôn ngữ tự nhiên.
Đối với các biểu diễn từ nhạy ngữ cảnh như GPT, biểu diễn của từ phụ thuộc vào ngữ cảnh của từ đó. Trong khi đó GPT không phân biệt tác vụ nhưng chỉ mã hóa ngữ cảnh theo chiều từ trái sang phải.
BERT kết hợp những gì tốt nhất của các mô hình trên: mã hóa ngữ cảnh theo hai chiều và chỉ yêu cầu những thay đổi kiến trúc tối thiểu cho một loạt các tác vụ xử lý ngôn ngữ tự nhiên.
Các embedding của chuỗi đầu vào BERT là tổng các embedding cho token, embedding đoạn và embedding vị trí.
Quá trình tiền huấn luyện BERT gồm có hai tác vụ: tác vụ mô hình hóa ngôn ngữ có mặt nạ và tác vụ dự đoán câu tiếp theo. Tác vụ đầu có thể mã hóa ngữ cảnh hai chiều để biểu diễn từ, trong khi tác vụ sau mô hình hóa mối quan hệ logic giữa các cặp văn bản một cách tường minh.
Kết quả dự đoán chưa quá cao. Các phương pháp ML vẫn thể hiện tốt trên tác vụ phân loại. Mô hình ngôn ngữ lớn DL chưa huấn luyện trước trên ngôn ngữ tiếng Việt vẫn chưa phân loại tốt. Mô hình PhoBert có kết quả tốt nhất trong các phương pháp biểu diễn ngữ cảnh output giúp cải thiện kết quả. Bước tiền xử lý dữ liệu là rất quan trọng.

Cải thiện và phát triển Sử dụng phương pháp tiền xử lý dữ liệu để cải thiện chất lượng dữ liệu. Thử nghiệm thêm các phương pháp biểu diễn ngữ cảnh khác của transformer

6 Đóng góp

6.1 Bùi Nguyễn Anh Trung

- Xác định bài toán, chọn bộ dữ liệu
- Chạy thực nghiệm dữ liệu trên các mô hình phân loại máy học, các mô hình ngôn ngữ lớn
- Tìm hiểu, trình bày nội dung kiến thức BERT và các phiên bản, biến thể của BERT
- Tìm hiểu, chạy thực nghiệm và đánh giá các cách tận dụng biểu diễn ngữ cảnh của Transformer
- Đóng góp làm slide và trình bày

6.2 Nguyễn Đặng Nhật Hào

- Phân tích, trình bày về bộ dữ liệu.
- Chạy thực nghiệm PhoBert và demo lên web.
- Đóng góp làm slide và báo cáo.

6.3 Huỳnh Chấn Huy

- Tìm hiểu về bộ dữ liệu
- Đề xuất và trình bày các thuật toán máy học.
- Tìm hiểu và trình bày các mô hình học sâu
- Đánh giá bộ dữ liệu và kết quả chạy thực nghiệm, đưa ra kết luận
- Đóng góp làm slide và báo cáo.

Table 3: Bảng phân công công việc

	Trung	Hào	Huy
Bộ dữ liệu	60	30	10
Mô hình máy học	40	10	50
Mô hình học sâu	60	30	10
Chạy thử nghiệm	60	30	10
Kết luận	40	20	40
Slide	40	20	40
Báo cáo	15	15	70

7 Tài liệu

https://towardsdatascience.com/bert-explained-state-of-the-art-language-mod
https://viblo.asia/p/hieu-hon-ve-bert-buoc-nhay-lon-cua-google-eW
https://huggingface.co/docs/transformers/model_doc/bert
https://d2l.aivivn.com/index.html