AUTOMATED ESSAY SCORING
HỆ THỐNG CHẤM ĐIỂM BÀI VĂN TỰ ĐỘNG

Kiều Sơn Tùng, Bùi Thị Thu Hương, Nguyễn Thị Hoài Linh, Trần Minh Quang

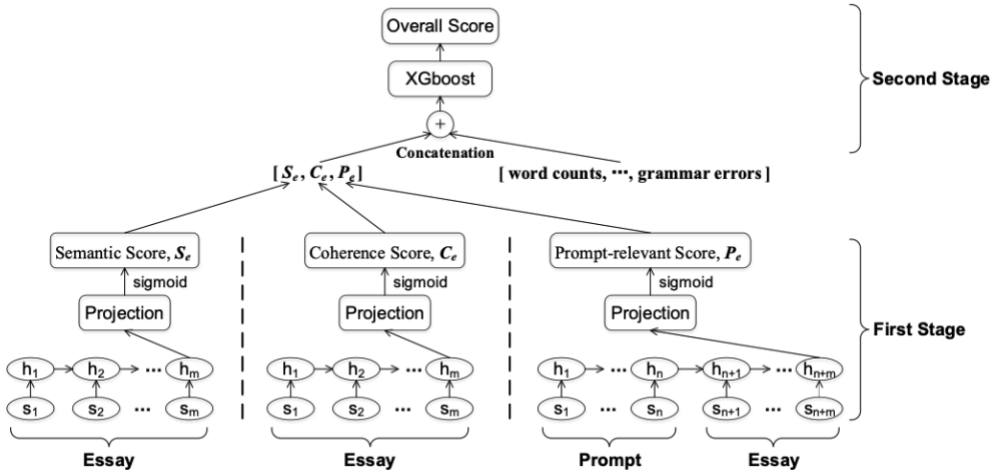
CHƯƠNG I: GIỚI THIỆU CHUNG VỀ NGHIÊN CỨU

1.1. Đặt vấn đề:

Việc đánh giá trong hệ thống giáo dục đóng một vai trò vô cùng quan trọng trong việc đáng giá năng lực của học sinh. Những hệ thống đánh giá hiện nay đều do con người thực hiện. Và với tỉ lệ số lượng giáo viên và học sinh ngày càng tăng lên, quá trình chấm điểm thủ công trở nên phức tạp hơn. Việc đáng giá năng lực bằng tay tốn nhiều thời gian, thiếu độ tin cậy và hơn thế nữa. Hệ thống kiểm tra tự động đã được phát triển như một công cụ thay thế cho các phương pháp truyền thống dựa trên giấy và bút. Hệ thống đánh giá bằng máy tính hiện tại chỉ hoạt động với những câu hỏi trắc nghiệm, nhưng không có hệ thống đáng giá nào đủ tốt để chấm điểm những đoạn văn hoặc những câu trả lời ngắn. Bài báo cáo này cung cấp một hệ thống đánh giá tài liệu và hệ thống chấm điểm bài luận tự động (AES – Automated Essay Scoring). Nhóm đã nghiên cứu các kỹ thuật Trí tuệ nhân tạo (Artificial Inteligence) và Máy học (Machine Learning) được sử dụng để đánh giá và cho điểm bài luận một cách tự động.

1.2. Tình hình nghiên cứu:

Đã xuất hiện nhiều những hệ thống chấm điểm luận văn tự động và một vài trong số chúng đã được sử dụng trong các bài đánh giá có rủi ro cao. Ở trong bài nghiên cứu về AES ở trường Đại học Khoa học Công nghệ ở Trung Quốc, một nhóm nghiên cứu đã phát triển được một thuật toán mới là TSLF (Two-Stage Learning Framework) đã tích hợp được lợi ích của phương pháp trích xuất đặc trưng (featured-engineered) và phương pháp end-to-end.



TSLF cho AES

Đầu tiên, dữ liệu sẽ được chia làm 3 nhóm chính:

- Coherence score (Ce) (điểm ngữ nghĩa): là độc lập nhanh chóng và được sử dụng để đánh giá các essay từ cấp độ ngữ nghĩa sâu sắc.
- Semantic Score (Se) (điểm mạch lạc): được khai thác để phát hiện các essay gồm các đoạn văn được hoán vị.
- Prompt-relevant score (Pe): mối liên hệ giữa prompts và essay được đánh giá dựa trên Prompt-relevant score, được dùng để phát hiện các mẫu prompt-irrelevant.

Sau đó, mô hình sẽ được chia làm 2 giai đoạn:

- First stage: Sử dụng Learning Framework(TSLF) , kết hợp những ưu điểm của các mô hình được thiết kế theo tính năng và mô hình end-to-end. Sau đó sẽ tính điểm semantic, coherence và prompt-relevant.
- Second stage: 3 điểm số này cùng với 1 số tính năng thủ công được nối và đưa vào boosting tree để training thêm

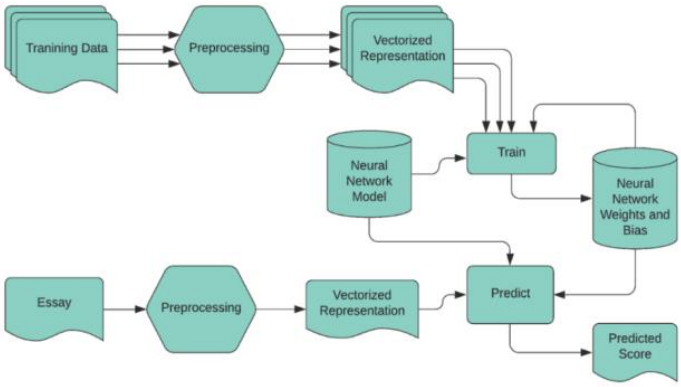
Kết quả dự đoán của mô hình đạt được 77,3% trên thang điểm Kappa, vượt trội so với đa số các AES khác sử dụng phương pháp Machine Learning hoặc Deep Learning cơ bản.

Trong một nghiên cứu khác của Bộ khoa học máy tính và kỹ thuật, Viện Khoa học và Khoa học S.R.M. Công nghệ Chennai, Ấn Độ, nhóm đã xây dựng một thuật toán cho điểm bài luận mạnh mẽ nhưng hiệu quả, tiêu chí là dựa trên thuộc tính của mô hình như sự trôi chảy của ngôn ngữ, tính đúng ngữ pháp và cú pháp, từ vựng và loại từ được sử dụng, đo độ dài bài luận, thông tin miền. Để trích xuất các đặc điểm này, nhóm đã sử dụng thư viện 'text mining', trích xuất ma trận tài liệu thuật ngữ để có thể cung cấp dữ liệu. Một số đặc điểm quan trọng khác để đánh giá bất kỳ đoạn văn nào ngoài nội dung thì số lượng từ trong các lớp cú pháp khác nhau như danh từ, trạng từ, động từ, tính từ cũng cần được đánh giá. Để có số lượng từ trong mỗi lớp POS(part-of-speech), nhóm nghiên cứu đã sử dụng thư viện NLTK. Ngoài ra, số lỗi chính tả trong một bài luận cũng được đưa vào như một đặc điểm để chấm điểm cho bài văn. Cuối cùng, nội dung của lĩnh vực là đặc điểm nổi bật nhất của mô hình vì nó cố gắng hiểu ngữ nghĩa và nội dung thông tin của một bài luận. Để tính năng này hoạt động, trước tiên, nhóm đã tìm ra bài luận hay nhất từ mỗi bộ (bài luận được điểm cao nhất), sau đó, rút ra các danh từ từ bài luận đó. Những danh từ này được dùng làm từ khóa cho lĩnh vực cụ thể. Sau đó, những từ này được đưa vào 'WordNet' và lấy ra từ tương đương của chúng. Bằng cách này, đối với mỗi nhóm, mô hình có một loạt các từ khác nhau, có liên quan nhất đến lĩnh vực cụ thể của nó. Sau đó, mô hình sẽ đếm số từ trong bài luận được cung cấp. Tuy nhiên khác với những bài nghiên cứu khác, kết quả của mô hình lại được nhóm đánh giá bằng định lý Bayes, đem lại độ chính xác 80%.

Wilon Zhu và Yu Sun đã đề xuất một mô hình nghiên cứu tạo ra AES sử dụng nhiều mô hình Machine Learning. Trong bài báo này, họ cố gắng kết hợp các phương pháp tiếp cận của feature extraction model và word vector model. Sử dụng feature extraction để có được số lượng từ, lỗi ngữ pháp và một phần của số lượng giọng nói và triển khai các word vector như GloVe, phương pháp này có thể đo lường cả các features dạng số của bài luận cũng như các features về ngữ cảnh và mức độ liên quan của nó với chủ đề. So với các phương pháp chỉ

có feature extraction hoặc word vector , phương pháp của họ kết hợp cả hai và tận dụng lợi thế của cả hai phương pháp trong khi giảm thiểu những thiếu sót của việc chỉ sử dụng một phương pháp. Do đó, việc chấm điểm bài luận tự động của họ có khả năng ít bị ảnh hưởng bởi các bài luận 'học để kiểm tra'.

Bài luận tự động bao gồm hai thành phần chính, tokenizer của bài luận và neural network model. Tokenizer chuyển đổi bài luận, một chuỗi, thành hai vectơ, một vectơ chứa biểu diễn số và một vectơ khác chứa biểu diễn word vector của bài luận. Sau đó, vectơ được chuyển vào neural network để đánh giá mô hình bằng cách sử dụngtrained model và trả về điểm số. Quá trình tiền xử lý xuất ra một biểu diễn số để feature extraction cũng như một chuỗi cho word vector model. Biểu diễn số bao gồm: đếm ngữ pháp, đếm số. Trình tự đầu vào cho mạng word vector neural network bao gồm một chuỗi có kích thước cố định từ tokenizer văn bản Keras. Tổng quan về giải pháp của họ được trình bày trong hình bên dưới:



Tổng quan về quá trình

Để tối đa hóa hiệu suất, họ quyết định kiểm tra hai tham số: loại neural network được sử dụng để nhúng GloVe và độ dài của kích thước nhúng GloVe. Các loại neural network mà họ đã thử nghiệm là Bộ nhớ ngắn hạn dài (Long Short-Term Memory), bộ nhớ định mức (Gated Recurrent Unit-GRU) và LSTM hai hướng (Bidirectional LSTM). Chiều dài khác nhau của kích thước nhúng GloVe là 50, 100, 200 và 300. Để kiểm traneural network tối ưu nhất, họ đã lấy điểm trung bình kappa trên các kích thước khác nhau của thứ nguyên nhúng GloVe của network. Mô hình hoạt động tốt nhất cho LSTM, GRU và Bidirectional LSTM thu được Kappa lần lượt là 070026, 0,68525 và 0,70024, với LSTM là neural network hoạt động tốt nhất nói chung. Để tìm thứ nguyên nhúng GloVe tối ưu nhất, họ đã so sánh quadratic weighted kappa của mỗi thứ nguyên nhúng với các neural networks khác nhau.

Dimension\Network	LSTM	GRU	BiLSTM	Average
50	0.70003	0.57328	0.70024	0.65785
100	0.68477	0.68525	0.67184	0.68062
200	0.70026	0.61821	0.68884	0.66910
300	0.69411	0.67478	0.63192	0.66694

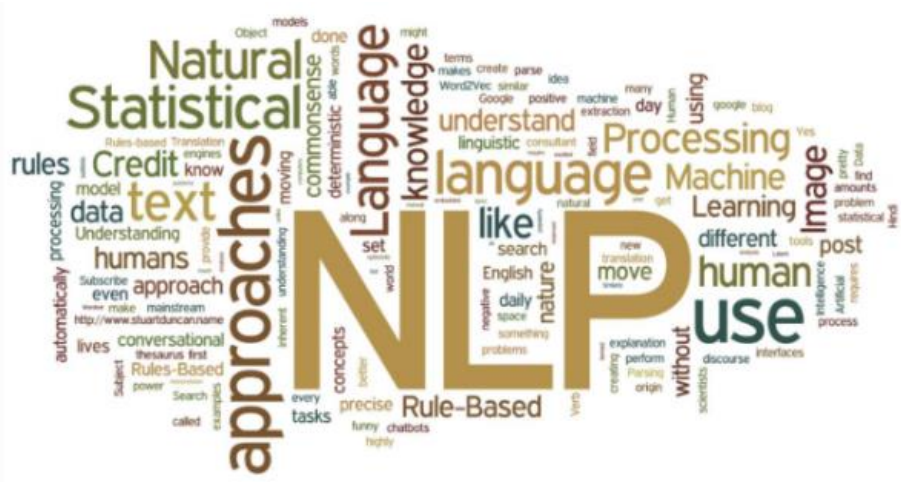
Chỉ số QWK của các mô hình

Từ bảng này, có thể thấy rằng kích thước nhúng không ảnh hưởng nhiều đến hiệu suất của mô hình LSTM. Tuy nhiên, thứ nguyên nhúng đã ảnh hưởng rất nhiều đến GRU và Bidirecitonal LSTM ở một mức độ thấp hơn.

CHƯƠNG II: CƠ SỞ LÝ THUYẾT

2.1. Xử lý ngôn ngữ tự nhiên NLP

Xử lý ngôn ngữ tự nhiên(gọi tắt NLP-Natural Language Processing) là một lĩnh vực nghiên cứu giúp máy tính hiểu, xử lý và nhận dạng các ngôn ngữ tự nhiên.Trọng tâm chính của nghiên cứu xử lý ngôn ngữ tự nhiên là dịch văn bản, trích xuất thông tin và tìm kiếm thông tin có liên quan.



Tổng quan các nghiên cứu về xử lý ngôn ngữ tự nhiên

2.1.1.Word2Vec

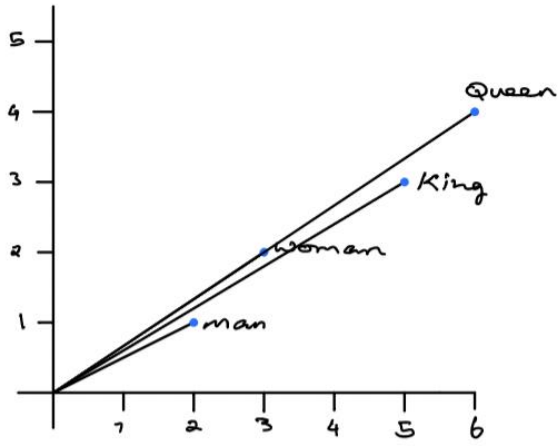
Word2vec là một mô hình đơn giản và nổi tiếng giúp tạo ra các biểu diễn embedding của từ trong một không gian có số chiều thấp hơn nhiều lần so với số từ trong từ điển. Ý tưởng của Word2vec đã được sử dụng trong nhiều bài toán với dữ liệu khác xa với dữ liệu ngôn ngữ.
Word2vec định nghĩa hai embedding vector cùng chiều cho mỗi từ *w* trong từ điển. Khi nó là một từ đích, embedding vector của nó là **u**; khi nó là một từ ngữ cảnh, embedding của nó là **v**. Sở dĩ ta cần hai embedding khác nhau vì ý nghĩa của từ đó khi nó là từ đích và từ ngữ cảnh là khác nhau. Tương ứng với đó, ta có hai ma trận embedding **U** và **V** cho các từ đích và các từ ngữ cảnh.

Hiệu quả của Word2Vec đến từ khả năng nhóm các vectơ của các từ tương tự lại với nhau. Với một tập dữ liệu đủ lớn, Word2Vec có thể đưa ra các ước tính mạnh mẽ về nghĩa của một từ dựa trên sự xuất hiện của chúng trong văn bản. Những ước tính này mang lại các liên kết từ với các từ khác trong kho ngữ liệu. Ví dụ, những từ như “King” và “Queen” sẽ rất giống nhau. Khi tiến hành các phép toán đại số trên các phép nhúng từ, bạn có thể tìm thấy một giá trị gần đúng của các từ tương tự. Ví dụ: vectơ embedding 2 chiều của "King" - vectơ embedding 2

chiều của "người đàn ông" + vectơ embedding 2 chiều của "phụ nữ" tạo ra một vectơ rất gần với vectơ embedding của "nữ hoàng". Lưu ý rằng các giá trị bên dưới được chọn tùy ý.

King - Man + Woman = Queen

[5,3] - [2,1] + [3, 2] = [6,4]



Có thể thấy 2 từ King và Queen rất gần với nhau về mặt vị trí.

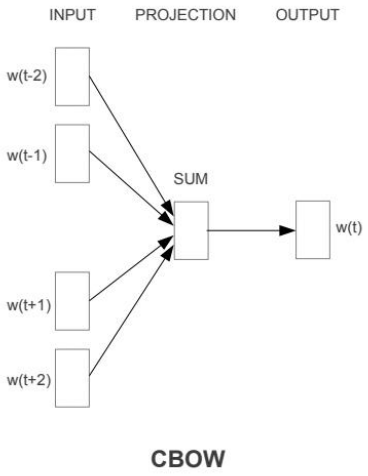
Có hai cách khác nhau xây dựng mô hình word2vec:

- Skip-gram: Dự đoán những từ ngữ cảnh nếu biết trước từ đích.
- CBOW (Continuous Bag of Words): Dựa vào những từ ngữ cảnh để dự đoán từ đích.

Mỗi cách có những ưu nhược điểm khác nhau và áp dụng với những loại dữ liệu khác nhau

2.1.1.1.CBOW (Continuous Bag of Words)

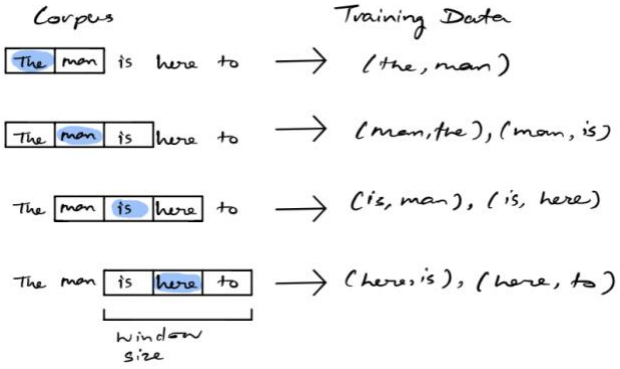
Kiến trúc này rất giống với một mạng nơ-ron chuyển tiếp. Kiến trúc mô hình này về cơ bản cố gắng dự đoán một từ đích từ danh sách các từ ngữ cảnh. Trực giác đằng sau mô hình này khá đơn giản: với một cụm từ "Have a great day", chúng ta sẽ chọn từ mục tiêu của mình là "a" và các từ ngữ cảnh của chúng ta là [“have”, “great”, “day”]. Những gì mô hình này sẽ làm là lấy các đại diện phân tán của các từ ngữ cảnh để thử và dự đoán từ đích.



Cấu trúc của CBOW

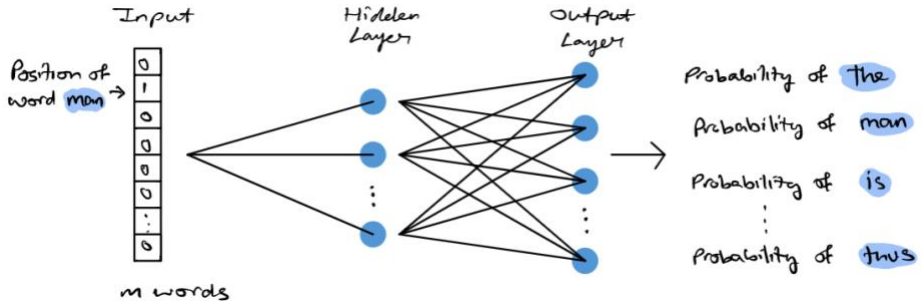
2.1.1.2.Continuous Skip-Gram Model

Mô hình skip-gram liên tục là một mạng nơ-ron đơn giản với một lớp ẩn được huấn luyện để dự đoán xác suất xuất hiện của một từ nhất định khi có từ đầu vào. Bằng trực giác, bạn có thể tưởng tượng mô hình skip-gram ngược lại với mô hình CBOW. Trong kiến trúc này, nó lấy từ hiện tại làm đầu vào và cố gắng dự đoán chính xác các từ trước và sau từ hiện tại này. Mô hình này về cơ bản cố gắng tìm hiểu và dự đoán các từ ngữ cảnh xung quanh từ đầu vào được chỉ định. Dựa trên các thí nghiệm đánh giá độ chính xác của mô hình này, người ta thấy rằng chất lượng dự đoán được cải thiện khi có nhiều vectơ từ, tuy nhiên nó cũng làm tăng độ phức tạp tính toán. Quá trình này có thể được mô tả trực quan như hình dưới đây.



Ví dụ về tạo dữ liệu đào tạo cho mô hình skip-gam. Kích thước window là 3.

Như đã thấy ở trên, với một số ngữ liệu văn bản, một từ đích được chọn trên một số cửa sổ cuốn. Dữ liệu đào tạo bao gồm các kết hợp theo cặp của từ đích đó và tất cả các từ khác trong cửa sổ. Đây là dữ liệu huấn luyện kết quả cho mạng nơ-ron. Một khi mô hình được đào tạo, về cơ bản chúng ta có thể mang lại xác suất một từ là một từ ngữ cảnh cho một mục tiêu nhất định. Hình ảnh dưới đây thể hiện kiến trúc của mạng nơ-ron cho mô hình skip-gam .



Mô hình Skip-Gram

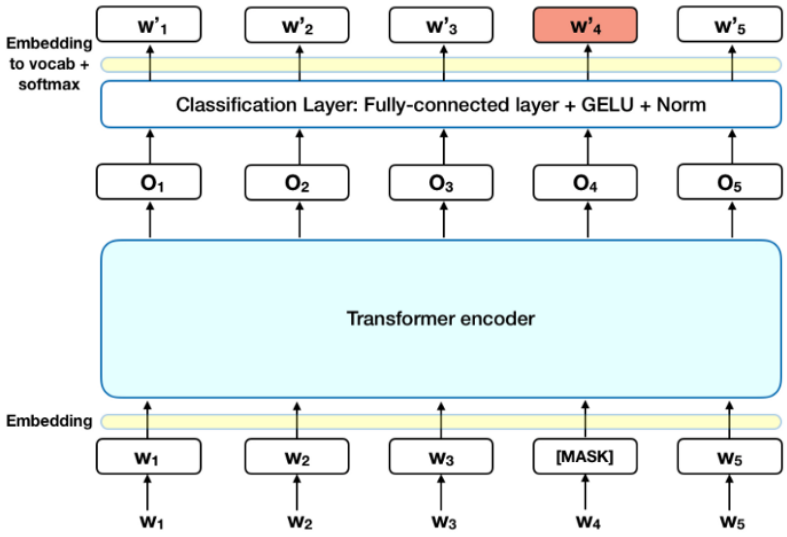
Một kho ngữ liệu có thể được biểu diễn dưới dạng một vectơ có kích thước N, trong đó mỗi phần tử trong N tương ứng với một từ trong kho ngữ liệu. Trong quá trình huấn luyện, chúng ta có một cặp từ đích và ngữ cảnh, mảng đầu vào sẽ có 0 trong tất cả các phần tử ngoại trừ từ đích. Từ đích sẽ bằng 1. Lớp ẩn sẽ học cách biểu diễn nhúng của mỗi từ, tạo ra không gian nhúng d-chiều. Lớp đầu ra là một lớp dày đặc có chức năng kích hoạt softmax. Lớp đầu ra về cơ bản sẽ mang lại một vectơ có cùng kích thước với đầu vào, mỗi phần tử trong vectơ sẽ bao gồm một xác suất. Xác suất này chỉ ra sự giống nhau giữa từ đích và từ được liên kết trong kho ngữ liệu.

2.1.2.BERT

BERT (Bidirectional Encoder Representations from Transformers) là một mô hình ngôn ngữ (Language Model) được tạo ra bởi Google AI. BERT được coi như là 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 (attention mechanism) học mối tương quan giữa các từ (hoặc 1 phần của từ) trong một văn bản. Transformer gồm có 2 phần chính: Encoder và Decoder, Encoder thực hiện đọc dữ liệu đầu vào và Decoder đưa ra dự đoán. Ở đây, bởi vì mục đích chính của BERT là tạo ra một language model, BERT chỉ sử dụng Encoder.

Khác với các mô hình directional (các mô hình chỉ đọc dữ liệu theo 1 chiều duy nhất - trái→phải, phải→ trái) đọc dữ liệu theo dạng tuần tự, Encoder đọc toàn bộ dữ liệu trong 1 lần, việc này làm cho BERT có khả năng huấn luyện dữ liệu theo cả hai chiều, qua đó mô hình có thể học được ngữ cảnh (context) của từ tốt hơn bằng cách sử dụng những từ xung quanh nó (phải & trái).



Mô hình encoder

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 ccs vector có kích thước đúng bằng kích thước input. Trong khi huấn luyện mô hình, một thách thức gặp phải là các mô hình directional truyền thống gặp giới hạn khi học ngữ cảnh của từ. Để khắc phục nhược điểm của các mô hình cũ, BERT sử dụng 2 chiến lược training như sau:

a. Masked 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]. Mask LM gồm các bước xử lý sau :

- Thêm một classification layer với input là output của Encoder.

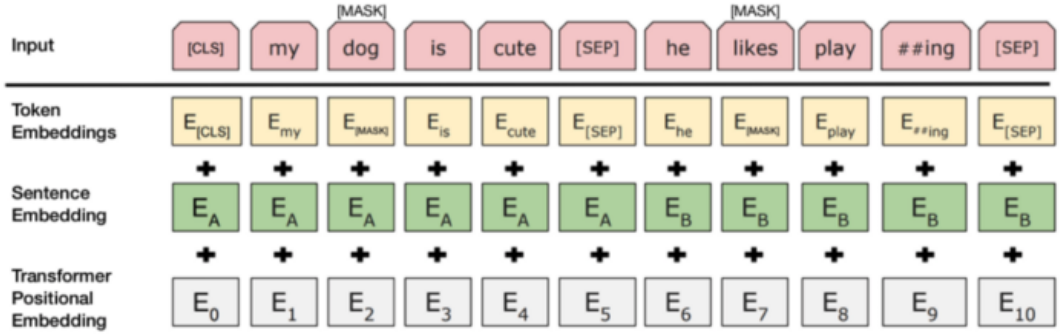
- Nhân các vector đầu ra với ma trận embedding để đưa chúng về không gian từ vựng (vocabulary dimensional).
- Tính toán xác suất của mỗi từ trong tập từ vựng sử dụng hàm softmax.

Hàm lỗi (loss function) của BERT chỉ tập trung vào đánh giá các từ được đánh dấu [MASKED] mà bỏ qua những từ còn lại, do đó mô hình hội tụ chậm hơn so với các mô hình directional, nhưng chính điều này giúp cho mô hình hiểu ngữ cảnh tốt hơn. (Trên thực tế, con số 15% không phải là cố định mà có thể thay đổi theo mục đích của bài toán.)

b. Next Sentence Prediction (NSP)

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. Một số nguyên tắc được đưa ra khi xử lý dữ liệu như sau:

- Chèn token [CLS] vào trước câu đầu tiên và [SEP] vào cuối mỗi câu.
- Các token trong từng câu được đánh dấu là A hoặc B.
- Chèn thêm vector embedding biểu diễn vị trí của token trong câu.



Next Sentence Prediction

Các bước xử lý trong Next Sentence Prediction:

- Toàn bộ câu đầu vào được đưa vào Transformer.
- Chuyển vector output của [CLS] về kích thước *2x1* bằng một classification layer.
- Tính toán xác suất *IsNextSequence* bằng softmax.
- Phương pháp Fine-tuning BERT

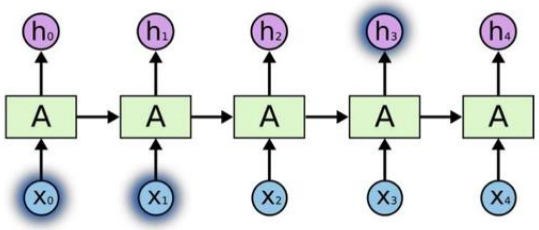
Tùy vào bài toán mà ta có các phương pháp fine-tune khác nhau:

- Đối với bài toán Classification, ta thêm vào một Classification Layer với input là output của Transformer cho token [CLS].
- Đối với bài toán 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 Named Entity Recognition (NER), model được huấn luyện để dự đoán nhãn cho mỗi token (tên người, tổ chức, địa danh,...).

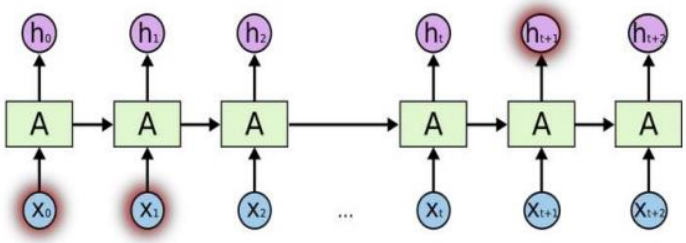
2.2. Mô hình đề xuất

2.2.1.RNN (Recurrent Neural Network)

Mạng nơ ron hồi quy RNN là một trong những mô hình deep learning. Recurrent có nghĩa là thực hiện lặp lại cùng một tác vụ cho mỗi thành phần trong chuỗi. Trong đó, kết quả đầu ra tại thời điểm hiện tại phụ thuộc vào kết quả tính toán của các thành phần ở những thời điểm đó. RNN là một mô hình có trí nhớ (memory), có khả năng nhớ được thông tin đã tính toán trước đó). Ý tưởng ban đầu của RNN là kết nối những thông tin trước đó nhằm hỗ trợ cho các xử lý hiện tại. Ví dụ, chúng ta dự đoán từ cuối cùng trong câu 'chuồn chuồn bay thấp thì mưa', thì chúng ta không cần truy tìm quá nhiều từ trước đó, ta có thể đoán ngay từ tiếp theo sẽ là 'mưa'. Trong trường hợp này, khoảng cách tới thông tin liên quan được rút ngắn lại, mạng RNN có thể học và sử dụng các thông tin trong quá khứ.



Trường hợp có nhiều thông tin hơn trong một câu, nghĩa là phụ thuộc vào ngữ cảnh. Ví dụ, khi dự đoán từ cuối cùng trong đoạn văn bản 'Tôi sinh ra và lớn lên ở Việt Nam. Tôi có thể nói thuần thục Tiếng Việt". Từ thông tin gần nhất cho thấy rằng từ tiếp theo là tên một ngôn ngữ. Nhưng khi chúng ta muốn biết cụ thể ngôn ngữ nào, thì cần quay về quá khứ xa hơn, để tìm được ngữ cảnh Việt Nam. Và như vậy, RNN có thể phải tìm những thông tin có liên quan và số lượng các điểm đó trở nên rất lớn.

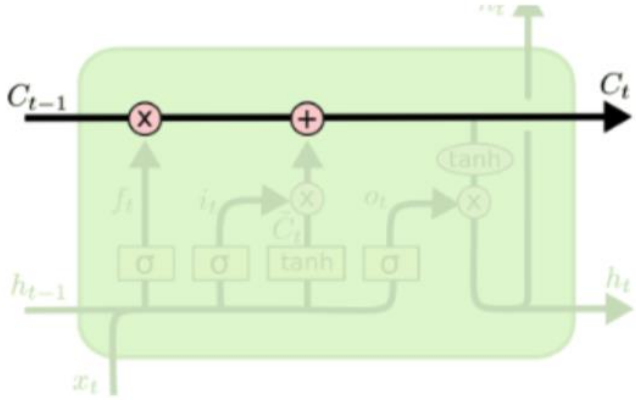


Về lý thuyết, RNN có thể nhớ được thông tin của chuỗi có chiều dài bất kỳ, nhưng trong thực tế mô hình này chỉ nhớ được thông tin ở vài bước trước đó.

2.2.2.LSTM (Long short term memory)

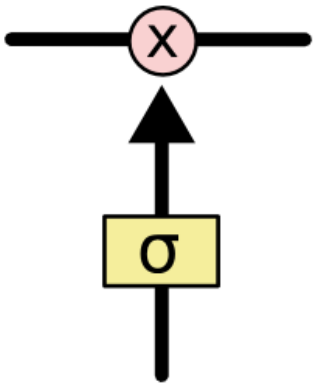
LSTM là một mạng thần kinh hồi quy (RNN) nhân tạo, được sử dụng trong lĩnh vực Deep Learning. LSTM chứa các kết nối phản hồi. Mạng không chỉ xử lý các điểm dữ liệu đơn lẻ, mà còn xử lý toàn bộ chuỗi dữ liệu (chẳng hạn như lời nói hoặc video). Bởi mạng RNN không thể ghi nhớ thông tin ở các bước có khoảng cách khá xa trước đó do vấn đề vanishing gradient. Do đó, những phần tử đầu tiên trong chuỗi đầu vào không có nhiều ảnh hưởng đến các kết quả tính toán dự đoán phần tử cho chuỗi đầu ra trong các bước sau. Mạng LSTM với các kết nối phản hồi (feedback connection) giúp khắc phục các nhược điểm này.

Ý tưởng cốt lõi của LSTM là trạng thái tế bào (cell state) - chính đường chạy thông ngang phía trên của sơ đồ hình vẽ. Trạng thái tế bào là một dạng giống như băng truyền. Nó chạy xuyên suốt tất cả các mắt xích (các nút mạng) và chỉ tương tác tuyến tính đôi chút. Vì vậy mà các thông tin có thể dễ dàng truyền đi thông suốt mà không sợ bị thay đổi.



Tế bào trạng thái LSTM giống như một băng truyền

LSTM có khả năng bỏ đi hoặc thêm vào các thông tin cần thiết cho trạng thái tế bào, chúng được điều chỉnh cẩn thận bởi các nhóm được gọi là cổng (gate). Các cổng là nơi sàng lọc thông tin đi qua nó, chúng được kết hợp bởi một tầng mạng sigmoid và một phép nhân.



Cổng trạng thái LSTM

Tầng sigmoid sẽ cho đầu ra là một trong số khoảng [0,1], mô tả có bao nhiêu thông tin có thể được thông qua. Khi đầu ra là 0 thì có nghĩ là không cho thông tin nào qua cả, còn khi là 1 thì có nghĩa là cho tất cả các thông tin đi qua nó. Một LSTM gồm có 3 cổng như vậy để duy trì và điều hành trạng thái của tế bào.

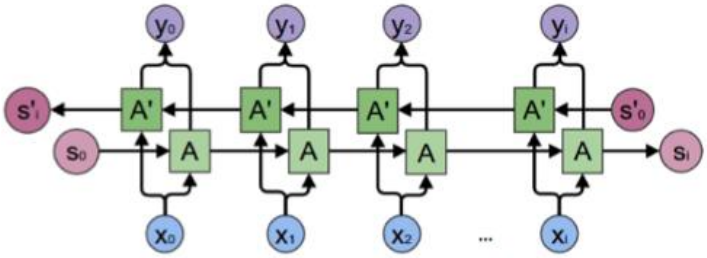
2.2.3.Bidirectional LSTM

Bidirectional LSTM cũng là một RNN chủ yếu được sử dụng để cho việc xử lý ngôn ngữ tự nhiên. Không như LSTM, đầu vào được chảy theo hai chiều, và nó có khả năng tối ưu thông tin từ cả hai phía.

BiLSTM thêm vào một lớp LSTM nữa để đảo ngược chiều của thông tin chảy vào. Điều đó có nghĩa là chuỗi đầu vào được chảy ngược lại vào lớp LSTM mới được thêm vào. Sau đó đầu ra sẽ được kết hợp bằng nhiều phương thức khác nhau như lấy tổng, trung bình hoặc móc nối với nhau.

Bidirectional LSTM là quá trình làm cho bất kỳ mạng nơ ron nào có thông tin trình tự theo cả hai hướng ngược (tương lai đến quá khứ) hoặc chuyển tiếp (quá khứ đến tương lai). Trong bidirectional (2 chiều), đầu vào của chúng chảy theo hai hướng, làm cho một Bi-LSTM khác với LSTM thông thường. Với LSTM thông thường, chúng ta có thể tạo luồng đầu vào theo

một hướng, ngược hoặc xuôi. Tuy nhiên, theo hai hướng, chúng ta có thể làm cho luồng đầu vào theo cả hai hướng để lưu giữ thông tin tương lai và quá khứ.
Do bản chất LSTM là cải tiến của RNN, cho nên ta có thể áp dụng nó và biến nó thành mạng nơ ron dài ngắn song song (Bi-LSTM). Mỗi LSTM vẫn có khả năng quên thông tin cũ (forget gate), lọc thông tin mới (input gate), hoặc giấu bớt kết quả (output gate) như bình thường. Chính vì vậy, các thông tin từ quá khứ tới tương lai của mạng BiLSTM đều có thể tự học để tự điều chỉnh. Dẫn tới việc với các bài toán mà ta cần biết nhiều hơn về ngữ cảnh hiện tại của nó, thì mạng BiLSTM cho kết quả tốt hơn.



Bidirectional LSTM

2.3. Phương pháp đánh giá

2.3.1.Quadratic Weighted Kappa

2.3.1.1.Định nghĩa:

Quadratic weighted Kappa là một chỉ số cho thấy sự đồng thuận giữa một tập hợp các dự đoán và một tập hợp các nhãn nhiều lớp. Ngoài việc chỉ đơn giản là xem xét độ chính xác phù hợp giữa các dự đoán và nhãn, nó còn cố gắng tính đến sự giống nhau giữa các lớp. Ngoài ra Quadratic weighted Kappa cũng tính đến khả năng xảy ra thỏa thuận một cách tình cờ, hoặc thỏa thuận ngẫu nhiên giữa các chuyên gia.

2.3.1.2.Các bước của QWK:

Đầu tiên, một ma trận biểu đồ O (NxN) được xây dựng sao cho $O_{i,j}$ tương ứng với số i (thực tế) nhận được giá trị dự đoán j. Ma trận NxN, trọng số(weighted), w, được tính toán dựa trên sự khác biệt giữa giá trị thực tế và giá trị dự đoán theo công thức:

$$w_{i,j} = \frac{(i-j)^2}{(N-1)^2}$$

Một ma trận biểu đồ NxN của các kết quả mong đợi (E), được tính toán với giả định rằng không có mối tương quan giữa các giá trị. Điều này được tính như tích ngoài giữa vector biểu đồ thực tế của kết quả và vector biểu đồ dự đoán, được chuẩn hóa sao cho E và O có tổng bằng nhau.

Cuối cùng, từ ba ma trận này, kappa có trọng số bậc hai được tính như sau

$$\boldsymbol{k} = 1 - \frac{\sum_{i,j} w_{i,j}\, O_{i,j}}{\sum_{i,j} w_{i,j}\, E_{i,j}}$$

Trong đó:

- w là ma trận có trọng số(weighted matrix)
- O là ma trận biểu đồ (histogram matrix)
- E là ma trận dự kiến (expected matrix)

2.3.1.3.Lý do lựa chọn Kappa:

Về ưu điểm, MSE đảm bảo cho mô hình được đào tạo không có dự đoán outlier với sai số lớn, vì MSE đặt trọng số lớn hơn vào các lỗi luận văn do phần bình phương của hàm. Nhược điểm là nếu mô hình đưa ra một dự đoán rất xấu, phần bình phương của hàm sẽ phóng đại sai số.

Trong phân loại, tập test và train cần chứa class tức là 0 hoặc 1 hoặc có thể là nhiều lớp hoặc tương tự. MSE là khoảng cách giữa thực tế (y_true) và dự đoán (y_pred). Trong bài này, có một số essay có cách tính điểm khá tương đối, ví dụ essay 5,6 domain_score có thể là min hoặc max hoặc trung bình hoặc một số gần với rate1_domain1 và rate2_domain1 (2 trường quan trọng dùng để tính domain1_score). Do đó, nếu dùng MSE thì có thể sẽ cho ra kết quả không tốt.

Ngoài ra, khi so sánh độ chính xác của các classifiers thường phải tính tới chi phí sai sót (cost of error). QWK, nó bù đắp cho những phân loại có thể do ngẫu nhiên. Việc sử dụng Weighted Kappa cho phép giải quyết hiệu quả việc phân loại nhạy cảm với chi phí (cost-sensitive).Khi chi phí sai sót không xác định được và chỉ có thể ước tính gần đúng, thì việc sử dụng phân tích độ nhạy (sensitivity) với Weighted Kappa sẽ tốt hơn.

2.4. Xử lý những đoạn văn bản dài

Trên thực tế, logic đằng sau việc tính toán cho các đoạn văn bản dài hơn là rất đơn giản. Nhóm sẽ lấy văn bản của mình (chẳng hạn như 1361 tokens) và chia nó thành nhiều phần chứa không quá 512 tokens mỗi loại.

Một tensor chứa 1361 tokens có thể được chia thành ba tensor nhỏ hơn. Hai dãy (list) đầu tiên sẽ chứa 512 tokens, với tensor cuối cùng chứa 337 mã còn lại. Sau đó nhóm đã chuyển đổi các phần khi lấy được và để chúng sẵn sàng được BERT tiêu thụ (sẽ sớm có thêm thông tin) - nhóm chuyển chúng qua mô hình của mình và truy xuất điểm cho mỗi phần. Cuối cùng, mức trung bình được lấy cho mỗi lớp - cung cấp cho bài dự đoán tổng thể cho toàn bộ đoạn văn bản (tất cả 1361 mã thông báo). Dưới đây là các bước nhóm đã thực hiện để tách những đoạn văn bản có hơn 512 tokens.

2.4.1.Initialization

Điều cần làm bây giờ là khởi tạo mô hình và trình mã hóa. Nhóm sẽ sử dụng PyTorch và thư viện máy biến áp Hugging Face cho cả 2 quá trình. Nhóm sẽ sử dụng mô hình BERT để phân loại trình tự và trình mã hóa BERT tương ứng.

2.4.2.Tokenization

Token hóa là quá trình chuyển đổi một chuỗi văn bản thành một danh sách các mã thông báo (các từ, dấu chấm câu riêng lẻ) hoặc ID mã thông báo (các số nguyên ánh xạ một từ với biểu diễn vectơ của từ đó trong một mảng embedding array). Các tham số này tạo nên cách tiếp cận

điển hình để mã hóa. Tuy nhiên, chúng chỉ đơn giản là không tương thích khi nhóm muốn chia một chuỗi dài hơn thành nhiều phần ngắn hơn.

Ngoài ra, các mã thông báo đặc biệt [CLS] và [SEP] được kỳ vọng lần lượt ở đầu và cuối của một chuỗi. Vì nhóm sẽ tạo các chuỗi này riêng biệt, nhóm cũng phải thêm các mã thông báo này một cách riêng biệt.

2.4.3.Preparing The Chunks

Bây giờ nhóm nghiên cứu đã có tensor mã hóa; nhóm nghiên cứu cần chia nó thành nhiều phần không quá 510 tokens. Nhóm chọn 510 thay vì 512 để chừa lại hai chỗ để thêm mã thông báo [CLS] và [SEP] của nhóm.

2.4.4.CLS và SEP

Tiếp theo, nhóm đã thêm mã bắt đầu của chuỗi [CLS] và dấu phân tách [SEP]. Mã thông báo đã ở định dạng mã thông báo ID, vì vậy nhóm đã có thể tham khảo bảng mã thông báo đặc biệt ở trên để tạo phiên bản mã thông báo cho mã thông báo [CLS] và [SEP] của bài nghiên cứu.

2.4.5.Padding

Nhóm cần thêm đệm vào các khối tensor của bìa để đảm bảo chúng thỏa mãn chiều dài 512 tensor theo yêu cầu của BERT. Để kiểm tra xem một đoạn có yêu cầu đệm hay không, ta kiểm tra độ dài tensor.

2.4.6.Reshaping for BERT

Trong bài, nhóm có các tensors, nhưng bây giờ cần định hình lại chúng thành các tensor đơn và thêm chúng vào từ điển đầu vào cho BERT. Sau đó, nhóm nghiên cứu định dạng chúng thành một từ điển đầu vào và thay đổi kiểu dữ liệu theo yêu cầu của BERT. Dữ liệu sẽ được chuyển vào BERT.

2.4.7.Making Predictions

Cuối cùng là đưa ra những dự đoán. Nhóm đã lấy một đoạn văn bản dài chứa hàng nghìn mã thông báo, chia nhỏ thành nhiều phần, thêm các mã thông báo đặc biệt và chia nhỏ trên tất cả các phần.

CHƯƠNG III: KẾT QUẢ MÔ HÌNH

Nhóm nghiên cứu đã đào tạo các mô hình học sâu (deep learning) một cách riêng biệt trên các tập riêng lẻ và trên toàn bộ tập dữ liệu. Nhóm đã sử dụng BERT và word2vec embedding cho mỗi tập dữ liệu này. Các giá trị được báo cáo là điểm QWK được tính 5 lần bằng phương pháp K-fold (5 folds). Nhóm đã đào tạo tất cả các mô hình bằng cách sử dụng tối ưu hóa (optimizer) Adam, mean-square-error là hàm phân tích tổn thất (loss function).

Kết quả mô hình sẽ được chia làm 2 phần: phần 1 sẽ báo cáo kết quả của mô hình khi sử dụng Word2Vec để tokenize những đoạn văn bản và sử dụng mô hình LSTM và Bidirectional LSTM

để dự đoán; phần 2 sẽ báo cáo kết quả mô hình sử dụng BERT để tokenize văn bản và cũng sử dụng LSTM và Bidirecitonal LSTM để dự đoán.

3.1. Word2Vec

Thực hiện việc huấn luyện dữ liệu trên mô hình với bộ dữ liệu huấn luyện. Chúng tôi sử dụng Tensorflow framework và các thư viện học sâu của Keras trong mô hình huấn luyện của mình.

Tất cả các LSTM neural networks được sử dụng trong bài của nhóm là một lớp (single-layer) với kích thước ẩn (hidden_dim) là 1024. Trên thực tế, nhóm cũng đã kiểm tra hiệu suất của mô hình LSTM nhiều lớp và chuyển hướng (multi-layer and BiLSTM). Nhưng kết quả không tốt như kỳ vọng. Để tránh overfitting quá nhiều, việc dropout được áp dụng trong quá trình đào tạo và tỷ lệ được đặt là 0.5.

Trong bảng báo cáo kết quả, cột MAPE của set 3, 4, 5 và 6 được để trống bởi các set này chứa điểm 0 nên không thể tính toán được chỉ số phần trăm sai số trung bình tuyệt đối của những set này. Do công thức tính chỉ số MAPE có mẫu số là giá trị thực tế của dữ liệu nên những bộ văn bản này có chứa điểm 0 không thể tính được.

3.1.1.Trên toàn bộ dataset

W2V + LSTM				W2V + LSTM			
QWK	MSE	MAE	MAPE (%)	QWK	MSE	MAE	MAPE (%)
0.0	126.758	6.802	96.777	0.0	126.759	6.802	96.777

Từ bảng kết quả trên ta có thể thấy rằng: việc huấn luyện mô hình trên toàn bộ dữ liệu cho ra kết quả không tốt. Word2Vec embedding cả hai mô hình LSTM và Bi LSTM cho kết quả gần như bằng nhau. Chỉ số MSE quá cao (hơn 126) và QWK thì lại quá thấp (bằng 0). Trong dữ liệu ASAP, tiêu chí chấm điểm và thang điểm của từng set là khác nhau. Do đó, huấn luyện trên toàn bộ dữ liệu sẽ cho kết quả không đúng và cho ra kết quả không tốt. Vì vậy, nhóm đã huấn luyện mô hình trên từng set một để phù hợp với thang điểm của từng set.

3.1.2.Trên từng set

	W2V + Bi LSTM				W2V + LSTM			
Set	QWK	MSE	MAE	MAPE	QWK	MSE	MAE	MAPE
1	0.546	3.595	1.428	18.744	0.335	4.418	1.637	19.856
2	0.458	0.967	0.708	24.552	0.438	0.929	0.694	23.626
3	0.479	1.160	0.820		0.479	1.145	0.813	
4	0.674	1.532	0.954		0.669	1.555	0.961	
5	0.619	1.610	0.975		0.627	0.594	1.587	
6	0.712	1.662	0.957		0.679	1.575	0.917	
7	0.563	30.085	4.394	33.589	0.468	30.963	4.495	31.442
8	0.488	42.989	5.122	14.753	0.444	44.353	5.212	14.644

Ta có thể thấy, việc huấn luyện mô hình theo từng set cho kết quả tốt hơn so với huấn luyện cả dataset. Nhìn chung, chỉ số QWK khi huấn luyện mô hình word2vec với BiLSTM cho ra chỉ số cao hơn so với LSTM. Đồng thời, nhìn chung chỉ số MSE và MAE của word2vec với BiLSTM cũng cho kết quả nhỏ hơn so với LSTM.

3.2. BERT

3.2.1.Chưa xử lý những đoạn văn bản có nhiều hơn 512 tokens

	BERT + LSTM				BERT + Bi LSTM			
Set	QWK	MSE	MAE	MAPE (%)	QWK	MSE	MAE	MAPE (%)
1	0.523	3.676	1.444	18.936	0.601	1.551	0.918	11.32
2	0.527	1.026	0.735	24.249	0.472	0.569	0.501	16.303
3	0.606	1.168	0.819		0.614	0.498	0.428	
4	0.775	1.604	0.976		0.756	0.416	0.388	
5	0.767	1.678	0.998		0.758	0.447	0.418	
6	0.722	1.652	0.944		0.712	0.513	0.460	
7	0.789	35.424	4.777	36.28	0.752	9.243	2.367	16.17
8	0	33.670	4.605	13.217	0.435	23.802	3.827	10.892

Với mô hình sử dụng là BERT cùng với LSTM, chỉ số QWK nằm trong khoảng 0.5 đến 0.8, duy chỉ có set 8 có chỉ số QWK bằng 0. Kết quả có được có lẽ bởi set 8 là set có ít văn bản nhất, với khoảng 700 bài văn nhưng thang điểm của set này trải dài từ 10 đến 60, từ đó dẫn tới phân phối điểm không đồng đều. Điều này dẫn tới việc mô hình không thể học được hết tính

chất đặc trưng của các đoạn văn trong các thang điểm khác nhau, như vậy sẽ dẫn tới tình trạng dự đoán nhãn bị lệch.

Có thể thấy rằng đa số các set khi được xử lý văn bản bằng BERT và đưa vào mô hình Bidirectional LSTM đều có chỉ số QWK nằm trong khoảng 0.6 đến 0.8, tuy nhỏ hơn so với mô hình LSTM nhưng cao hơn so với những nghiên cứu cùng đề tài. Chỉ số QWK của set 2 và set 7 sử dụng mô hình LSTM có cao hơn so với việc sử dụng mô hình Bidirectional LSTM nhưng phần trăm sai số lại cao hơn, điều đó chỉ ra rằng mô hình LSTM đang bị quá khớp với tập huấn luyện. Chỉ số MAE từ set 1 đến set 6 đều ở mức 0.3 đến 0.9 điểm, điều này là chấp nhận được khi việc chấm điểm của những bài văn còn tùy thuộc vào cảm nhận của từng giám khảo. Ở set 7 và set 8, do khoảng điểm của 2 set này là khá lớn, lần lượt là 2 đến 24 và 10 đến 60 điểm nên sự sai số trung bình tuyệt đối của 2 set này cao. Chỉ số MAPE của các set đều nằm trong khoảng từ 6% đến 17%, sở dĩ có sự sai số lớn như vậy ở một vài set là do sự phân phối điểm không đồng đều của các set từ đó dẫn đến sự mất cân bằng trong dữ liệu, gây nhiễu cho mô hình.

3.2.2.Đã xử lý những đoạn văn bản có hơn 512 tokens

	BERT +Bi LSTM			
Set	QWK	MSE	MAE	MAPE (%)
1	0.783	0.877	0.625	7.406
2	0.587	0.409	0.380	12.120
3	0.650	0.415	0.369	
4	0.633	0.514	0.462	
5	0.762	0.408	0.377	
6	0.606	0.582	0.480	
7	0.640	11.65	2.67	20.107
8	0.437	24.568	3.868	10.913

Sau khi đã xử lý những đoạn văn bản có trên 512 tokens, kết quả của một số set có được cải thiện đáng kể. Tiêu biểu nhất là chỉ số QWK của set 1 đã tăng từ 0.601 lên 0.783, chỉ số MAPE cũng giảm từ 11.32% xuống 7.406%. Một số set khác như set 2, 3, 5, 8 có sự tăng nhẹ trong độ chính xác so với khi chỉ tokenize văn bản rồi cho vào mô hình để huấn luyện. Tuy nhiên, set 4, 6, và 7 chứng kiến chỉ số QWK giảm nhẹ khoảng 10% so với giá trị trước khi xử lý những đoạn văn bản dài. Điều này có thể do những set 4, 6, 7 không có nhiều đoạn văn bản quá dài nên không được hưởng lợi từ phương pháp xử lý như những set khác. Có nhiều những vector 0 (padding) có thể gây nhiễu cho mô hình, tác động không tốt đến kết quả dự đoán.

CHƯƠNG IV: KẾT LUẬN

Trong dự án này, nhóm nghiên cứu đã đề xuất một phương hướng tiếp cận để giải quyết nhiệm vụ chấm điểm bài văn tự động dựa trên hai phương thức tokenizer là BERT và Word2Vec và xây dựng mô hình dựa trên cấu trúc LSTM và Bidirectional LSTM. Ngoài ra, nhóm nghiên

cứu còn đề xuất phương pháp để giải quyết với những đoạn văn dài, cụ thể là những văn bản có trên 512 kí tự. Nhờ có phương pháp xử lý này mà mô hình đã có sự cải thiện đáng kể độ chính xác trong các set, chỉ số QWK tăng từ 5% đến 30% so với mô hình không xử lý những văn bản dài, các chỉ số đo lường sự sai số của mô hình như MSE, MAE và MAPE giảm từ 25% đến 34% .

Tuy nhiên, bài nghiên cứu vẫn gặp một vài những hạn chế. Đầu tiên là bộ dữ liệu chưa thực sự tốt. Các set văn bản trong bộ dữ liệu có dải điểm không thống nhất cũng như phân phối dữ liệu chưa thực sự đồng đều khiến cho mô hình không thể học hết được đặc điểm, thuộc tính của những bài văn trong từng thang điểm từ đó chấm điểm một cách chính xác nhất. Và hệ thống chấm điểm bài luận được sử dụng bộ dữ liệu Kaggle ASAP (2012), bộ dữ liệu này có các bài luận chung của sinh viên và không yêu cầu bất kỳ lĩnh vực cụ thể nào, vì vậy cần có bộ dữ liệu bài luận theo đề tài cụ thể để huấn luyện và kiểm tra. Trích xuất đặc trưng bằng các thư viện NLTK, WordVec và BERT có nhiều hạn chế trong khi chuyển đổi một câu thành dạng vector, câu có thể không mang đúng và đủ như ý nghĩa ban đầu. Khi các câu được chuyển từ chữ thành số, rất khó để con người có thể can thiệp và điều chỉnh vector đó theo ý muốn của mình.

Trong tương lai, nhóm sẽ nghiên cứu để tìm ra một bộ dữ liệu sạch hơn, có một chủ đề nhất định và thang điểm phù hợp hơn, từ đó có thể giúp dự đoán chính xác hơn. Ngoài ra, nhóm nghiên cứu sẽ thử nghiệm một số phương pháp tokenize mới hiện đại được huấn luyện với nhiều dữ liệu hơn và công phu hơn, từ đó quá trình chuyển hóa từ văn bản sang vector sẽ sát nghĩa với văn bản hơn.

TÀI LIỆU THAM KHẢO

1. Hewlett (2012). *The Hewlett Foundation: Automated Essay Scoring*. [online] kaggle.com. Available at: https://www.kaggle.com/competitions/asap-aes/data .

2. Ke, Z. and Ng, V. (n.d.). *Automated Essay Scoring: A Survey of the State of the Art*. [online] Available at: https://www.ijcai.org/proceedings/2019/0879.pdf.

3. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R. and Research, G. (n.d.). *Published as a conference paper at ICLR 2020 ALBERT: A LITE BERT FOR SELF-SUPERVISED LEARNING OF LANGUAGE REPRESENTATIONS*. [online] Available at: https://openreview.net/pdf?id=H1eA7AEtvS .

4. Liu, J., Xu, Y., Zhu, Y. and Fopure (n.d.). *Automated Essay Scoring based on Two-Stage Learning*. [online] Available at: https://arxiv.org/pdf/1901.07744v2.pdf.

5. Mikolov, T., Chen, K., Corrado, G. and Dean, J. (2013). *Efficient Estimation of Word Representations in Vector Space*. [online] Available at: https://arxiv.org/pdf/1301.3781.pdf.

6. Nguyễn Văn, T. (2019). *UBND TỈNH BÌNH DƯƠNG TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT NGUYỄN TRUNG TÍN XÂY DỰNG HỆ THỐNG HỎI ĐÁP TỰ ĐỘNG HỖ TRỢ CÔNG TÁC TƯ VẤN DỊCH VỤ HÀNH CHÍNH CÔNG TẠI SỞ THÔNG TIN VÀ TRUYỀN THÔNG TỈNH BÌNH DƯƠNG LUẬN VĂN THẠC SĨ CHUYÊN NGÀNH: HỆ THỐNG THÔNG TIN MÃ SỐ: 8480104 BÌNH DƯƠNG -2019*. [online]
Available at: http://viewer.tdmu.edu.vn/wpViewFile.aspx?EdataFileDetailId=14138.

7. Ramesh, D. and Sanampudi, S.K. (2021). An automated essay scoring systems: a systematic literature review. *Artificial Intelligence Review*. doi:10.1007/s10462-021-10068-2.

8. Sanh, V., Debut, L., Chaumond, J. and Wolf, T. (2019). *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. [online] arXiv.org. Available at: https://arxiv.org/abs/1910.01108.

9. V. V. Ramalingam, A Pandian, Prateek Chetry and Himanshu Nigam (2018). *Automated Essay Grading using Machine Learning Algorithm*. [online] Available at: https://iopscience.iop.org/article/10.1088/1742-6596/1000/1/012030/pdf.

10. Zhu, W. and Sun, Y. (2020). Automated Essay Scoring System using Multi-Model Machine Learning. *Computer Science & Information Technology (CS & IT)*. [online] doi:10.5121/csit.2020.101211.

11. Nirthika, R., Manivannan, S. and Ramanan, A. (2020). *Loss functions for optimizing Kappa as the evaluation measure for classifying diabetic retinopathy and prostate cancer images*. [online] IEEE Xplore. doi:10.1109/ICIIS51140.2020.9342711.

Code và data dùng để thực hiện báo cáo trên được đăng tải ở trong link này.

ĐÓNG GÓP

1. Kiều Sơn Tùng: 40%
2. Bùi Thị Thu Hương: 25%
3. Nguyễn Thị Hoài Linh: 20%
4. Trần Minh Quang: 15%