ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

Soft-Masked XLM-R và Hard-Masked XLM-R cho bài toán sửa lỗi chính tả tự động

Huỳnh Nhật Hào

TP. HỒ CHÍ MINH – 1/2021

Contents

1. Giới Thiệu

Bài toán sửa lỗi chính tả tự động là một bài toán có tính ứng dụng cao trong thực tế. Sửa lỗi chính tả tự động có thể giúp người tự học ngôn ngữ biết được lỗi sai khi viết, hoặc trong trường hợp cần kiểm tra lỗi chính tả tự động cho các văn bản quan trọng nhưng dài, khiến cho việc kiểm tra thủ công trở nên mất thời gian và dễ nhầm lẫn.

Input của bài toán này sẽ là một câu có độ dài tùy ý có từ sai chính tả hoặc không, bao gồm cả từ viết tắt, teencode,… hoặc có từ không phù hợp trong ngữ cảnh câu nói, đi qua mô hình sửa lỗi sẽ phát hiện ra vị trí của từ sai và đề xuất một từ mới phù hợp với ngữ cảnh dựa vào các từ còn lại. Ví dụ:

Input: Tôi vẫn luôn iu cô ấy với hết tấm lòng của mk.

Output: Tôi vẫn luôn yêu cô ấy với hết tấm lòng của mình.

Với một điều kiện là tỉ lệ từ sai trong câu là khoảng 15%, vì mô hình đề xuất từ vào chỗ sai là mô hình *XLMRobertaForMaskedLM*[1] đã được huấn luyện sẵn, và mô hình này cần có ngữ cảnh để dự đoán từ thay thế.

Ngoài ra, mô hình được đề xuất còn có thể dự đoán được lỗi từ đúng chính tả nhưng không phù hợp ngữ cảnh, hoặc vô nghĩa khi đưa vào ngữ cảnh câu. Ví dụ:

Input: Những ngày cuối tuần đặc biệt của Bitcoin trong thời gian mèo đây đang đặt ra thử thách nước cho những cây chơi tiền điện tử, dù lớn hay nhỏ.

Output: Những ngày cuối tuần đặc biệt của Bitcoin trong thời gian gần đây đang đặt ra thử thách mới cho những người chơi tiền điện tử, dù lớn hay nhỏ.

Đồ án này trình bày hai phương pháp để giải bài toán sửa lỗi chính tả tự động cho tiếng Việt gồm: Soft-Masked XLM-R và Hard-Masked XLM-R. Cả hai phương pháp này đều dựa vào mô hình ngôn ngữ XLM-RoBERTa[2] do Facebook phát triển. Trong đó, phương pháp Soft-Masked XLM-R chỉ dừng lại ở nghiên cứu lý thuyết vì không có đủ tài nguyên

[1] https://huggingface.co/transformers/model_doc/xlmroberta.html#xlmrobertaformaskedlm
[2] Unsupervised Cross-lingual Representation Learning at Scale, Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, Veselin Stoyanov

tính toán để huấn luyện, còn phương pháp Hard-Masked XLM-R đã được thực nghiệm và cho kết quả rất tốt trong việc phát hiện ra lỗi sai trong câu và thay thế bằng một từ thích hợp.

Toàn bộ source code để chạy lại các thí nghiêm trong đồ án này đều được public tại https://github.com/huynnhathhao/vietnamese_text_correction

2.Phương Pháp

2.1. Giới thiệu XLM-R

XLM-RoBERTa là một mô hình đa ngôn ngữ được Facebook phát triển năm 2019 dựa trên đàn anh của nó là BERT hay Bidirectional Encoder Representation from Transformers[3]. Các mô hình ngôn ngữ này dùng kiến trúc Encoder của mô hình Transformers được giới thiệu trong bài báo Attention is all you need[4], ban đầu được phát triển cho bài toán Machine Translation. Mô hình được huấn luyện trên rất nhiều dữ liệu văn bản, giúp nó có kiến thức về ngôn ngữ cụ thể mà nó được huấn luyện trên đó với mục đích cuối cùng là dùng nó để tạo ra các vector Embedding đại diện cho các văn bản phục vụ cho các bài toán xử lý ngôn ngữ tự nhiên phía sau nó, như Sentiment analysis, Machine Translation, Question Answering, …

XLM-R được chọn cho bài toán Sửa lỗi chính tả tự động là vì nó là một trong những mô hình ngôn ngữ hiếm hoi được huấn luyện trên tiếng Việt. Trong các mô hình ngôn ngữ có hỗ trợ tiếng Việt phải kể đến PhoBERT[5], Multilingual BERT, XLM-R. Tuy nhiên XLM-R mặc dù là mô hình đa ngôn ngữ, được huấn luyện trên 100 ngôn ngữ trong đó có tiếng Việt, lại là mô hình được huấn luyện trên nhiều dữ liệu tiếng Việt nhất (khoảng 137GB tiếng việt, trong khi PhoBERT là mô hình một ngôn ngữ nhưng chỉ được huấn luyện trên 20GB tiếng Việt). Ngoài ra, XLM-R dựa trên mô hình RoBERTa[6] là đàn em của BERT nên đã có nhiều cải tiến trong việc huấn luyện mô hình để làm tăng khả năng diễn đạt ngôn ngữ bằng vector.

[3] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova
[4] Attention Is All You Need, Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin
[5] PhoBERT: Pre-trained language models for Vietnamese, Dat Quoc Nguyen, Anh Tuan Nguyen
[6] RoBERTa: A Robustly Optimized BERT Pretraining Approach, Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, Veselin Stoyanov

2.2. Soft-Masked XLM-R

Soft-Masked XLM-R được phát triển dựa trên mô hình Soft-Masked BERT[7], là một mô hình sửa lỗi chính tả tự động cho tiếng Trung Quốc xây dựng trên BERT. Cách hoạt động của Soft-Masked XLM-R hoàn toàn giống với Soft-Masked BERT, chỉ khác ở mô hình ngôn ngữ mà nó dùng để biểu diễn ngôn ngữ bằng các vector.

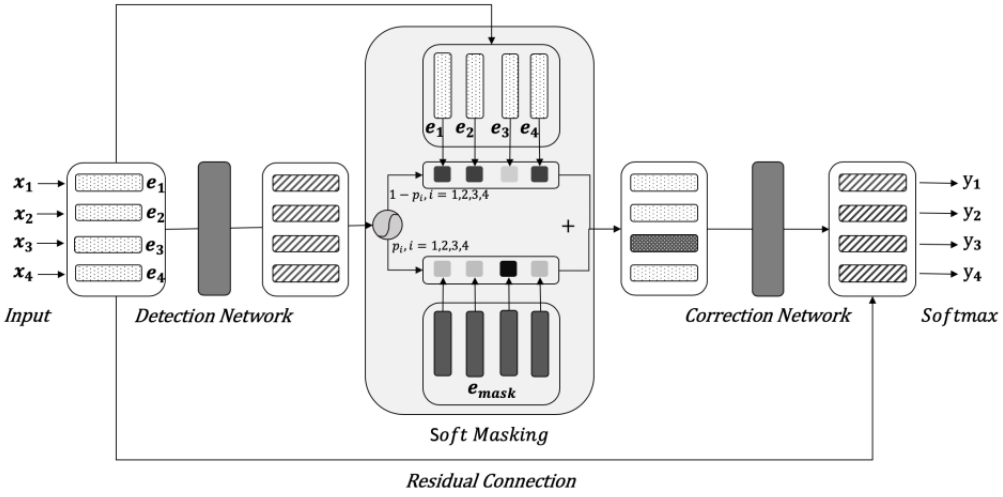


Figure 1: Architecture of Soft-Masked BERT

Hình 1: Soft-Masked BERT

Hình 1 mô tả cách hoạt động của Soft-Masked BERT từ input là một câu cho đến output là câu đó được sửa lỗi (nếu có).

Mô hình này được tạo thành từ 2 mạng: Detector network và Corrector network. Mạng Detector có nhiệm vụ tìm ra vị trí của từ bị sai chính tả, và dùng thông tin đó chuyển qua mạng Corrector. Mạng Corrector sẽ sử dụng thông tin vị trí từ bị sai cùng với các từ làm ngữ cảnh để tìm ra từ thích hợp nhất điền vào vị trí từ sai.

Câu input sẽ được tokenize dựa vào một tokenizer nào đó, mà thông thường số lượng token sẽ nhiều hơn số lượng từ trong câu ban đầu. Trong bài BERT, các tác giả sử dụng mô hình WordPiece Tokenizer, là một subword tokenizer, nó sẽ chia một từ ra thành các subword dựa vào dữ liệu mà nó được huấn luyện.

[7] Spelling Error Correction with Soft-Masked BERT, Shaohua Zhang, Haoran Huang, Jicong Liu, Hang Li

Tất cả các token sau đó đi qua mô hình embedding và trở thành một tensor có kích thước (Tx, N) trong đó Ty là số lượng token có trong câu và N là số chiều của một vector embedding. Mạng Detector nhận các vector embedding này và cho ra (Tx, 1) xác suất mà nó cho rằng một token thuộc từ bị sai chính tả. Mạng Detector của Soft-Masked BERT là một BiGRU.

Sau đó, các vector embedding mới sẽ được tạo dựa vào các vector embedding của các token ban đầu, các xác suất được Detector cho ra và vector embeding của <mask> token. Cụ thể,

new_embedding = p*(<mask> embedding) + (1-p)*embedding (1)

Trong đó p là output của Detector, <mask> embedding là vector embedding của <mask> token đã được huấn luyện từ trước của BERT, embedding là các vector embedding ban đầu đi vào mạng Detector. Các vector embedding mới này sẽ là input cho mạng Corrector.

Vector <mask> embedding là một vector đặc biệt đã được huấn luyện trước của mô hình BERT. Vector này tồn tại là do mô hình BERT được huấn luyện trên một task gọi là Masked Language Modeling. Với task này, mô hình sẽ phải dự đoán một từ bị che bởi token <mask> dựa vào các từ còn lại làm ngữ cảnh, và tỉ lệ từ bị mask là 15%.

Ví dụ:

Input*:* Không có <mask> tôi vẫn sống hạnh phúc.

Output: Không có em tôi vẫn sống hạnh phúc.

Vector embedding của <mask> token có tính chất đặc biệt là nó sẽ thông báo cho mô hình cần thay thế nó bởi một từ khác nằm trong từ vựng. Mạng Corrector đã lợi dụng tính chất này của <mask> token embedding để làm cho BERT dự đoán ra từ cần thay thế vào từ bị sai. Tuy nhiên khác với Hard-Masked, Soft-Masked không hoàn toàn thay thế một token mà nó cho rằng là từ sai chính tả bằng <mask> embedding mà chỉ lấy xác suất Detector cho rằng token đó là sai chính tả nhân với <mask> embedding bằng công thức (1).

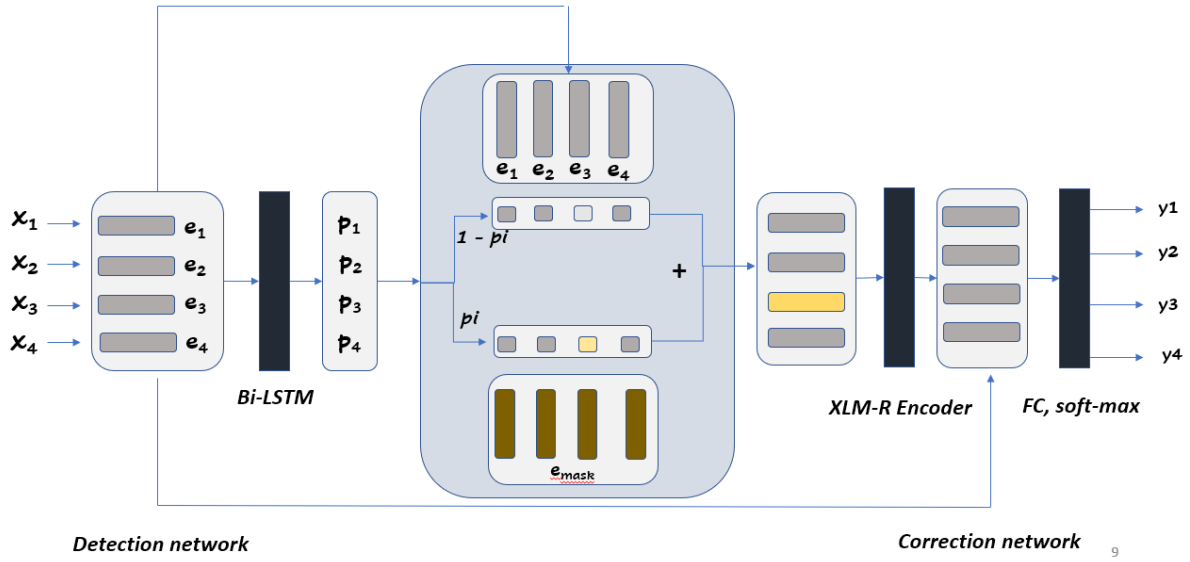
Mạng Corrector là phần Encoder của BERT. Khi nhận input là các vector embedding nó sẽ cho ra các vector mới, có kích thước giống như kích thước của input embedding (Tx, N), gọi là last hidden state. Các vector mới này đi qua một mạng Fully connected với output là số từ vựng trong từ điển của tokenizer và một hàm softmax sẽ cho ra xác suất trên toàn bộ từ vựng với xác suất của từ được thay thế là cao nhất (nếu có từ sai) và giữ nguyên các từ đúng

Tổng số lượng tham số của mô hình vừa mô tả có thể lên đến hàng trăm triệu, với phần lớn tham số tập trung vào phần Encoder của BERT và phần mạng Fully connected cuối cùng. Số lượng tham số của lớp FC cuối phụ thuộc vào số lượng từ vựng của tokenizer mà mô hình ngôn ngữ đó sử dụng. BERT tokenizer có 30000 từ vựng, còn XLM-R có 250002 từ vựng. Cả hai mô hình đều có số chiều của vector hidden state ở lớp FC là 768.

Model	Encoder	Last FC
Soft-Masked BERT	85M	23M
Soft-Masked XLM-R	270M	190M

Bảng 1: Số lượng tham số của Encoder và lớp FC cuối của Soft-Masked BERT và Soft-Masked XLM-R

Cộng thêm các tham số ở lớp Embedding và Detector thì Soft-Masked XLM-R có khoảng 550M tham số. Việc huấn luyện và fine-tune trên mô hình này cần rất nhiều tài nguyên tính toán. Đó là lý do chính mà mô hình Soft-Masked XLM-R chỉ dừng lại ở mức nghiên cứu lý thuyết mà chưa thể thực nghiệm để xem kết quả.



Hình 2: Soft-Masked XLM-R

Hình 2 mô tả kiến trúc mạng Soft-Masked XLM-R, trong đó Detector là một mạng BiLSTM và corrector là Encoder của XLM-R. Cách hoạt động của kiến trúc này giống như của Soft-Masked BERT.

2.3. Hard-Masked XLM-R

Hard-Masked XLM-R ra đời để giải quyết vấn đề về tài nguyên tính toán của Soft-Masked XLM-R. Hard-Masked XLM-R chỉ cần huấn luyện mạng Detector, và mạng Corrector là một mô hình XLMRobertaForMaskedLM[8] đã được huấn luyện sẵn để dự đoán <mask> token trong câu. Detector này sẽ là một mạng Bi-LSTM với hidden size là 512, embedding size là 512 và có 2 lớp. Tổng số tham số của Detector là khoảng 15M. Nhiệm vụ của Detector là dự đoán ra xác suất một token thuộc/là một từ sai chính tả, sau đó lấy ngưỡng 0.5 để quyết định một token có là từ sai chính tả hay không, với > 0.5 nghĩa là có và <= 0.5 là không, sau đó từ bị cho là sai chính tả sẽ bị thay thế bởi <mask>, các từ khác được giữ nguyên. Câu mới tạo thành sẽ đưa cho mạng Corrector và mạng Corrector có trách nhiệm đề xuất từ mới phù hợp vào chỗ bị sai chính tả.

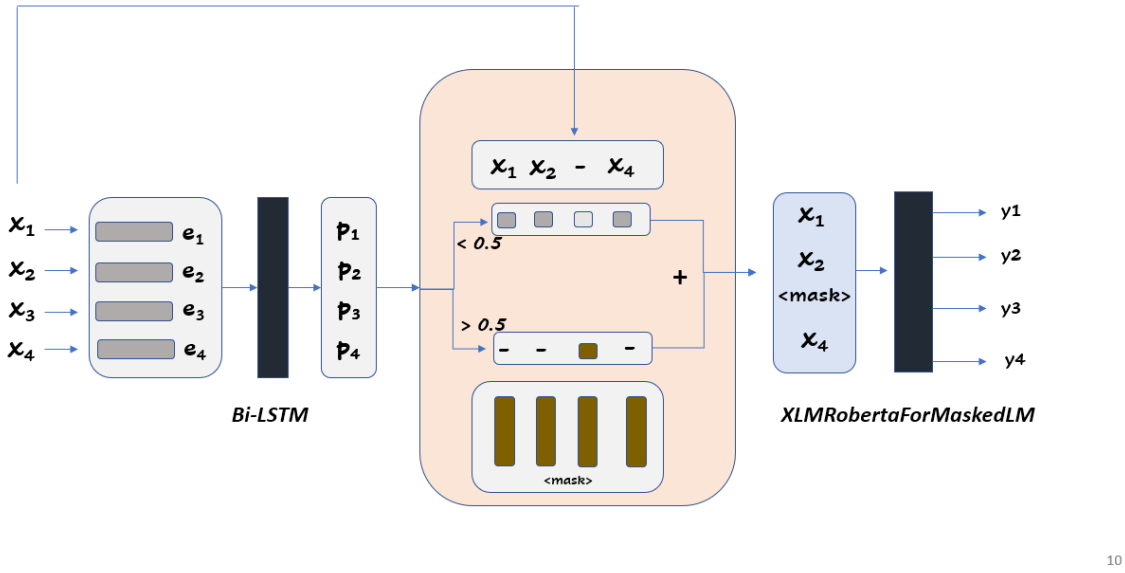
[8] https://huggingface.co/transformers/model_doc/xlmroberta.html#xlmrobertaformaskedlm

Ví dụ:

Input: Hàn Quốc dỡ bo một chatbot thịnh hành do phát ngôn thù ghét

Output: Hàn Quốc dỡ <mask> một chatbot thịnh hành do phát ngôn thù ghét

Câu mới này sẽ trở thành input cho mạng Corrector.



Hình 3: Kiến trúc mạng Hard-Masked XLM-R

Các đề xuất của mạng Corrector có thể đi qua một hàm tính khoảng cách giữa từ đề xuất và từ sai chính tả bằng edit distance, sau đó chọn ra từ đề xuất có khoảng cách edit distance ngắn nhất làm từ thay thế cho từ sai chính tả. Tuy nhiên trong phạm vi đồ án này, hàm tính khoảng cách chưa được cài đặt.

2.4. So sánh Soft-Masked và Hard-Masked

- Về số lượng tham số: Mô hình Hard-Masked tuy không cần phải huấn luyện Corrector, nhưng về tổng thể thì Hard-Masked lại có nhiều tham số hơn Soft-Masked. Đó là vì Hard-Masked tách rời mạng Detector của nó thành một phần riêng lẻ, trong khi Soft-Masked tích hợp Detector của nó vào trong mô hình. Số

lượng tham số của Hard-Masked nhiều hơn Soft-Masked bằng với số lượng tham số của mạng Detector bị tách rời ra.

- Về cách hoạt động:
    - Hard-Masked cần phải tokenize và embedding câu input đến 2 lần. Cụ thể là, mô hình Detector của Hard-Masked sử dụng một tokenizer dựa vào SentencePiece tokenizer[9] được huấn luyện trên toàn bộ dữ liệu training và có 10000 từ vựng. Detector của Hard-Masked có một lớp Embedding với số chiều là 512, cả 2 phần tokenize và embedding này chỉ phục vụ cho mạng Detector. Mạng Corrector sẽ lặp lại 2 bước tokenization và Embedding với tokenizer và lớp Embedding của riêng nó. Việc lặp lại cùng một công việc 2 lần này làm tiêu tốn thêm nhiều tài nguyên tính toán và bộ nhớ.
    - Soft-Masked vì tích hợp cả phần Detector vào mô hình nên chỉ cần tokenize và embedding câu input chỉ một lần.

3. Tạo dữ liệu huấn luyện

Vì tiếng Việt hiện chưa có bộ dữ liệu benchmark nào cho bài toán Sửa lỗi chính tả tự động, nên toàn bộ dữ liệu huấn luyện và đánh giá đều được tạo thông qua một hàm tổng hợp. Hàm này nhận vào một câu tiếng Việt đúng chính tả, tạo ra một hay nhiều lỗi sai với tỉ lệ 15% với độ dài của câu. Các lỗi sai được dựa vào các từ đồng âm, các chữ cái đồng âm như s, x, … các từ viết tắt như không: k, ko, hk, hong, …. Ngoài ra để tạo dữ liệu về các lỗi đánh máy, các từ sai được tạo bằng cách thêm/ xóa /thay thế một chữ cái ngẫu nhiên trong một từ ngẫu nhiên được chọn trong câu, tạo thành các lỗi nonword. Các lỗi realword được tạo bằng cách thay thế một từ ngẫu nhiên trong câu thành một từ ngẫu nhiên trong từ điển, không thích hợp với ngữ cảnh của câu. Việc huấn luyện trên dữ liệu này sẽ giúp Detector có khả năng phát hiện các lỗi là một từ có trong từ điển nhưng không phù hợp với ngữ cảnh trong câu nói. Hàm tổng hợp sẽ output ra câu có từ sai chính tả, nhãn one hot chỉ vị trí của từ sai và câu gốc đúng chính tả.

[9] SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing, Taku Kudo, John Richardson

Dữ liệu trong bài này được crawl từ các trang báo vietnamnet.vn, thanhnien.vn, vnexpress.net là các trang báo nổi tiếng ở Việt Nam. Tổng cộng sau khi crawl và loại bỏ các câu dư thừa, quá ngắn thì thu được hơn 3 triệu câu tiếng Việt thuộc các chủ đề khác nhau như chính trị, kinh tế, xã hội, văn hóa, …

Do tính chất ngẫu nhiên trong việc tạo lỗi của hàm tổng hợp, nên một câu khi đi qua hàm tổng hợp 2 lần sẽ cho ra 2 sample khác nhau với loại lỗi và vị trí lỗi khác nhau. Nên các câu tiếng Việt có thể đi qua hàm tổng hợp nhiều lần để tạo ra nhiều dữ liệu huấn luyện.

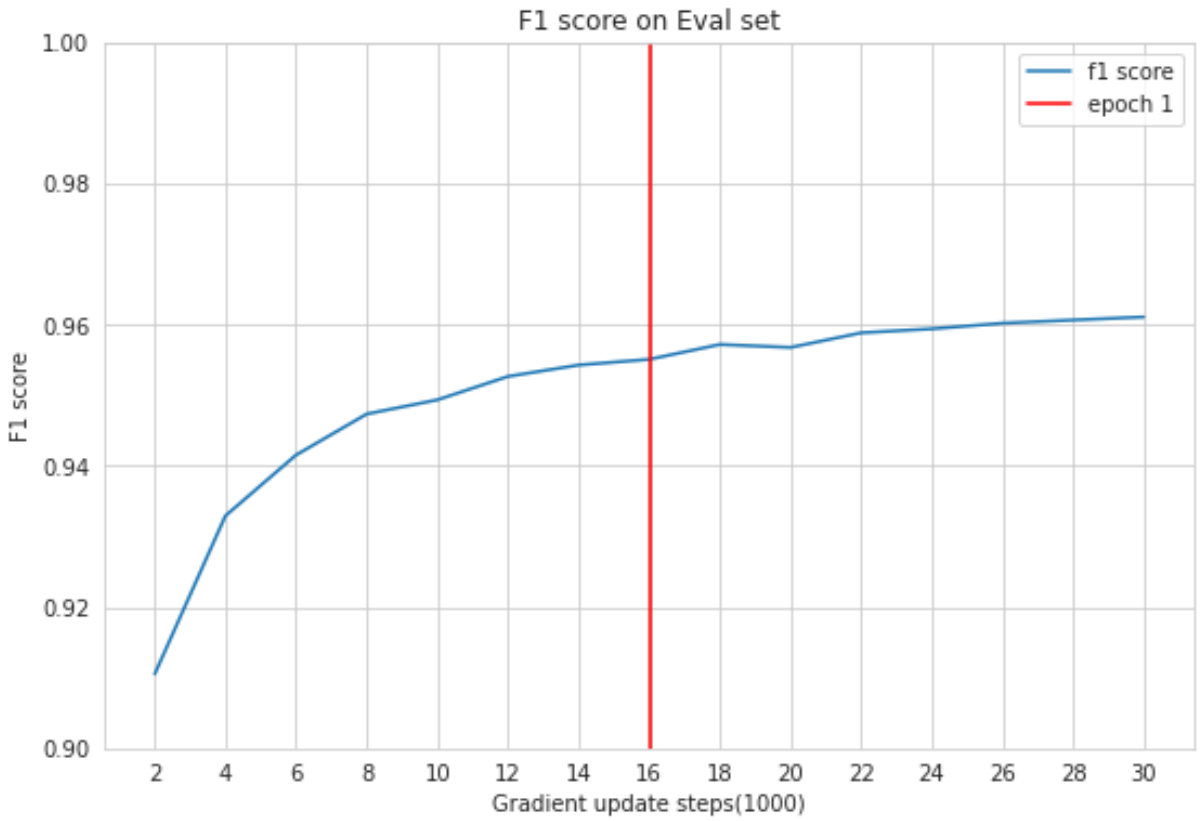
4. Kết quả thực nghiệm

4.1. Kết quả trên tập dev

Các kết quả quan sát được sau đây là của mô hình Hard-Masked XLM-R.

Tổng cộng dữ liệu thu được từ các trang web là hơn 3M câu tiếng Việt.

Detector được huấn luyện trên 2M câu tiếng việc bị giới hạn với độ dài tối đa là 100 tokens theo SentencePiece tokenizer. Tỉ lệ các loại lỗi trong 2M câu là 0.3 random_word, 0.2 random_letter, 0.1 homophone_word, 0.2 remove_diacritics, 0.2 homophone_char. Các tham số huấn luyện mô hình: Batch size: 128, learning_rate: 0.001, epochs: 2. Hình 4 cho thấy f1 score qua các bước cập nhật đạo hàm trên tập eval có 20 000 samples ngẫu nhiên trích từ 3M câu ban đầu, trong đó cứ 1 sau một batch 128 samples sẽ tính đạo hàm và cập nhật trọng số mô hình một lần. Tổng cộng là 30 000 lần cập nhật trọng số trong 2 epochs. F1 score chỉ tăng mạnh trong epoch đầu tiên và không còn tăng nhiều khi đạt 24000 lần cập nhật trọng số. Sau khi huấn luyện với learning rate 0.001 30 000 lần học, thì chuyển sang học với learning rate 0.0005 và kết quả cao nhất của Detector là 0.967 f1 score.



Hình 4: F1 score của Detector trong quá trình huấn luyện.

Đối với Corrector, vì chỉ sửa những lỗi nào đã được Detector chỉ ra nên bị phụ thuộc hoàn toàn vào Detector. Các từ mà Corrector đề xuất để sửa lỗi sai hoặc là khớp với từ của câu gốc ban đầu, hoặc là một từ khác phù hợp với ngữ cảnh nên không có phép đánh giá nào thể hiện tốt được khả năng của Corrector. Ví dụ, nếu Corrector nhận vào câu input:

Năm nay, các đường bay quốc <mask> dừng hoạt động do ảnh hưởng của dịch Covid-19 nhưng nhu cầu tăng cao nên cung vẫn không đủ cầu.

Corrector sẽ có đề xuất vào từ bị mask là ‘gia’ hoặc ‘tế’, vì quốc gia, quốc tế đều phù hợp trong ngữ cảnh câu này. Một hàm post process có thể dùng để chọn ra từ gần với từ sai chính tả để làm từ thay thế.

Một nhược điểm Detector thể hiện trong quá trình thử nghiệm là nó sẽ chỉ ra cả những từ không bị sai chính tả thành từ sai chính tả với tỉ lệ thấp, nếu từ đó là một từ ít gặp, hoặc có một từ khác mà nó cho là phù hợp hơn khi đứng ở đó. Điều này có thể là do trong tập huấn luyện của Detector có chứa các lỗi là các từ random không phù hợp ngữ

cảnh với tỉ lệ 0.3 tổng số samples. Để giải quyết vấn đề này, hoặc là bỏ hoàn toàn các sample chứa lỗi là từ random, hoặc là huấn luyện Detector trên nhiều sample có chứa từ random hơn để nó có thể nhận ra các trường hợp từ ít gặp nhưng vẫn đúng trong ngữ cảnh đó. Ví dụ:

Input:

Nhu cầu tăng cao nhưng cung không đủ thì không thể có vé giá rẻ. Tuy bay quốc tế dừng hoạt động nhưng nhu cầu đi đến các điểm du lịch trong dịp Tết cũng tăng hơn so với các năm trước nên lượng người đi lại cao điểm Tết rất đông", đại diện một hãng hàng không cho biết

Output:

Nhu cầu tăng cao nhưng nếu không đủ thì không thể có vé giá rẻ Tuy bay quốc tế vẫn hoạt động nhưng nhu cầu đi đến các điểm du lịch trong dịp Tết cũng tăng hơn so với các năm trước nên lượng người đi lại cao điểm Tết rất đông", đại diện một hãng hàng không cho biết

Ở ví dụ này, từ 'cung' đã bị thay thế bởi từ 'nếu', là do Detector cho rằng từ cung là không nên nằm trong ngữ cảnh câu nói, điều này là do Detector vẫn chưa nhìn thấy từ cung trong lúc huấn luyện nên nó không biết từ cung có nên nằm trong trường hợp này hay không. Nhược điểm này hoàn toàn có thể khắc phục bằng cách huấn luyện trên nhiều dữ liệu đa dạng hơn nữa.

4.2. Kết quả trên văn bản thực tế

Mô hình đã được chạy thử trên một bài viết của confession uit với nhiều từ viết tắt, tiếng anh,… để kiểm tra khả năng sửa lỗi của nó trong thực tế. Các emoji đã được loại bỏ. Sau đây là văn bản gốc, với các từ viết sai được gạch dưới và tô màu:

Chào bạn, mình hiện là sinh viên năm 3 thì năm nhất mình cũng có thời gian tồi tệ tương tự bạn. Kì 2 năm nhất thì sau khi thi giữa kì xong thì mình cũng out khỏi trường luôn (vì lí do mình bị phát bệnh Viêm Cột Sống Dính Khớp), trong thời gian đó thì mình gần như liệt tại giường ở phòng ktx, trãi qua những cơn đau về tinh thần lẫn thể

xác. Khi đó mình chẳng đủ sức lực để lo lắng cho bản thân, nhưng mình k dám nói cho người nhà vì mình mồ côi cha mẹ chỉ còn 2 em là người thân ruột thịt và các cô, bác thôi nên mình sợ 2 em biết thì lo lắng, còn cô bác thì phiền họ. Mình cảm thấy mình là gánh nặng, mình không nên sinh ra trên đời này, lúc đó mình chỉ ước mình biến mất khỏi thế giới và ai cũng sẽ không có kí ức về mình để họ không còn buồn khi mình ra đi. Còn với bạn bè thì mình lúc đó mình chả tin ai hết vì mình nghĩ con người mà đâu ai cho không ai cái gì mà đi giúp mình với suy nghĩ đó mình bắt đầu co mình lại và trốn tránh mọi người. Nhưng mình may mắn hơn bạn là anh ở chung phòng ktx rất tốt, ảnh đã giúp đỡ mình sinh hoạt trong những ngày đó ví dụ nguyên ngày mình không ăn gì thì cũng mua dùm mình cái bánh ngọt do ảnh là người khuyết tật á nên giúp vậy là ghê lắm r nha, rồi khuyên mình gọi về cho cô mặc dù mình chẳng nghe,...(còn nhiều nữa nhưng mình kể nhiêu thôi) Rồi một ngày kia khoảng cuối tháng 6 thì cô mình có gọi và mình đã khai thật mọi chuyện, lúc đó mọi người đã đưa mình đi bệnh viện pla pla, trong thời gian đi gần hết bệnh viện ở SG thì mình đc quen với nhiều người hơn, nhận được rất nhiều sự giúp đỡ của các anh chị, cô chú mặc dù chỉ là người dưng để rồi mình hiểu suy nghĩ của mình khi đó nó sai tới nhường nào, để mình biết xung quanh chúng ta luôn còn rất nhiều người tốt. Bây giờ mình đã quay lại với trường UIT thân yêu, và bắt đầu lại cuộc sống mới mặc dù tính vẫn còn hơi hướng nội tí, nên bạn ơi nếu bạn cảm thấy câu chuyện của mình có 1 chút nào đồng điệu với chuyện của bạn thì hãy mở lòng, và cố gắng react haha vào bài này nha, có lẽ mình sẽ giúp bạn 1 chút gì đó như lắng nghe chuyện của bạn,... chớ khuyên nhủ thì mình k chắc tại mình cũng hơi nhát, nhưng biết đâu bạn sẽ k thấy lạc lõng và tìm lại đc chính mình á

Lời cuối mong bạn có suy nghĩ tích cực lên, không học thì kiếm gì khác làm như đọc sách,... Hồi đó mình đọc truyện á, nhưng toàn truyện buồn nên bạn né tụi nó ra nha.

Chúc bạn sớm vực dậy và tìm lại đc ý nghĩa của cuộc sống.

Có tổng cộng 13 lỗi, bao gồm cả từ viết tiếng anh, từ viết tắt trong bài viết trên.

Toàn bộ bài viết được đưa qua mô hình theo từng câu đã được chia lại, sau đây là kết quả của mô hình theo từng câu, các từ được sửa đúng tô màu xanh dương, từ sửa sai hay không sửa thì tô màu đỏ:

Chào bạn, mình hiện là sinh viên năm 3 thì năm nhất mình cũng có thời gian tồi tệ tương tự bạn.

Kể 2 năm nhất thì sau khi thi giữa kỳ xong thì mình cũng ra khỏi trường luôn (vì lí do mình bị phát bệnh Viêm Cột Sống Dính Khớp), trong thời gian đó thì mình gần như liệt tại giường ở phòng ktx, trãi nghiệm những cơn đau về tinh thần lẫn thể xác.

Khi đó mình chẳng đủ sức lực để lo lắng cho bản thân, nhưng mình k dám nói cho người nhà vì mình mồm cha mẹ chỉ còn 2 em là người thân ruột thịt và các cô, bác thôi nên mình sợ 2 em biết thì lo lắng, còn cô bác thì phiền họ.

Mình cảm thấy mình là gánh nặng, mình không nên sinh ra trên đời này, lúc đó mình chỉ ước mình biến mất khỏi thế giới và ai cũng sẽ không có kí ức của mình để họ không còn buồn khi mình ra đi.

Còn với bạn bè thì mình lúc đó mình không tin ai hết vì mình nghĩ con người mà đâu ai cho không ai cái gì mà đi giúp mình với suy nghĩ đó mình bắt đầu dựng mình lại và trốn tránh mọi người.

Nhưng mình may mắn hơn bạn là anh ở chung phòng mình rất tốt, ảnh đã giúp đỡ mình sinh hoạt trong những ngày đó ví dụ nguyên ngày mình không ăn gì thì cũng mua dùm một cái bánh ngọt do ảnh là người khuyết tật ái giúp vậy là ghê lắm rùi rồi khuyên mình gọi về cho cô mặc dù mình chẳng nghe, (còn nhiều nữa nhưng mình kể hết thôi).

Rồi một ngày kia khoảng cuối tháng 6 thì cô mình có gọi và mình đã khai thật mọi chuyện, lúc đó mọi người đã đưa mình đi bệnh viện rồi.

trong thời gian đi gần hết bệnh viện ở đây thì mình đến với nhiều người hơn, nhận được rất nhiều sự giúp đỡ của các anh chị, cô chú mặc dù chỉ là người dưng để cho

mình hiểu suy nghĩ của mình khi đó nó sai đến nhường nào, để mình biết xung quanh chúng ta luôn còn rất nhiều người tốt.

Bây giờ mình đã quay lại với trường học thân yêu, và bắt đầu lại cuộc sống mới mặc dù tính vẫn còn hơi hướng nội tí.

nên bạn, nếu bạn cảm thấy câu chuyện của mình có 1 chút nào đồng điệu với chuyện của bạn thì hãy mở lòng, và cố gắng lắng đọc bài này nha, có lẽ mình sẽ giúp bạn 1 chút gì đó như lắng nghe chuyện của bạn.

chớ khuyên nhủ thì mình, chắc tại mình cũng hơi nhát, nhưng biết đâu bạn sẽ k bị lạc lõng và tìm lại đc chính mình á. Lời cuối mong bạn có suy nghĩ tích cực lên, không học thì kiếm gì khác làm như đọc sách,... Hồi đó mình đọc truyện á, nhưng toàn vì buồn nên bạn bỏ tụi nó ra nha.

Chúc bạn sớm vực dậy và tìm lại được ý nghĩa của cuộc sống.

Đây là một văn bản khó so với mô hình, bởi vì toàn bộ dữ liệu huấn luyện của mô hình đều mang phong cách ngôn ngữ báo chí, không có các từ như ‘á’, ‘nha’, hay các từ tiếng anh lẫn lộn, nên mô hình sẽ gặp khó khăn khi không biết phải làm gì với các từ này. Tuy nhiên Detector vẫn nhận ra các lỗi sai thông thường và sửa được các từ như đc, kì, r(rồi), r(ra). Điều đặc biệt là chữ ‘r’ là chữ viết tắt của ‘rồi’ và ‘ra’ nhưng mô hình biết dựa vào ngữ cảnh để chọn ra từ đúng.

Giải pháp cải thiện mô hình phát hiện và sửa các lỗi bên trên là huấn luyện mô hình trên nhiều dữ liệu mang phong cách ngôn ngữ sinh hoạt.

5. Kết luận

Trong nội dung đồ án này, em đã nghiên cứu hai mô hình giải bài toán sửa lỗi chính tả tự động cho tiếng Việt là Soft-Masked XLM-R và Hard-Masked XLM-R, trong đó chỉ có mô hình Hard-Masked XLM-R được đưa vào thực nghiệm vì không đủ tài nguyên tính toán để huấn luyện Soft-Masked XLM-R. Kết quả thu được trên Hard-Masked XLM-R với khả năng nhận diện lỗi sai là 96.7%, tuy nhiên chỉ trên các văn bản mang

phong cách ngôn ngữ báo chí. Mô hình này có thể khắc phục được các lỗi không nhận diện được từ sai trong văn bản mang phong cách ngôn ngữ khác bằng cách huấn luyện nó trên các loại văn bản đó.